

Uncertainty In Artificial Intelligence

Proceedings of the Thirtieth Conference (2014)

Edited by

Nevin L. Zhang

Jin Tian

Uncertainty in Artificial Intelligence

Proceedings of the Thirtieth Conference (2014)

July 23-27, 2014, Quebec City, Quebec, Canada

Edited by

Nevin L. Zhang, The Hong Kong University of Science and
Technology, China

Jin Tian, Iowa State University, USA

General Chair

Ann Nicholson, Monash University/Bayesian Intelligence,
Australia

Sponsored by

Microsoft Research, Artificial Intelligence Journal, Facebook Inc.,
Google Inc., Charles River Analytics, IBM Research

AUAI Press Corvallis, Oregon

Cover design © Alice Zheng.

Published by AUI Press for
Association for Uncertainty in Artificial Intelligence
<http://auai.org>

Editorial Office:
P.O. Box 866
Corvallis, Oregon 97339
USA

Copyright © 2014 by AUI Press
All rights reserved
Printed in the United States of America

No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means—electronic, mechanical, photocopying, recording, or otherwise—without the prior written permission of the publisher.

ISBN 978-0-9749039-1-0

Contents

Preface	vii
Organizing Committee	ix
Acknowledgments	xi
Sponsors	xvii
Best Paper Awards	xix
1 Proceedings	1
MEMR: A Margin Equipped Monotone Retargeting Framework for Ranking. <i>Sreangsu Acharyya, Joydeep Ghosh</i>	1
On Convergence and Optimality of Best-Response Learning with Policy Types in Multiagent Systems. <i>Stefano Albrecht, Subramanian Ramamoorthy</i>	12
Accelerating MCMC via Parallel Predictive Prefetching. <i>Elaine Angelino, Eddie Kohler, Margo Seltzer, Amos Waterland, Ryan Adams</i>	22
A variational approach to stable principal component pursuit. <i>Aleksandr Aravkin, Stephen Becker, Volkan Cevher, Peder Olsen</i>	32
Markov Network Structure Learning via Ensemble-of-Forests Models. <i>Eirini Arvaniti, Manfred Claassen</i>	42
Can(Plan)+: Extending the Operational Semantics of the BDI architecture to deal with Uncertain Information. <i>Kim Bauters, Weiru Liu, Jun Hong, Carles Sierra, Lluís Godó</i>	52
Message Passing for Soft Constraint Dual Decomposition. <i>David Belanger, Alexandre Passos, Sebastian Riedel, Andrew McCallum</i>	62
Bayesian Interactive Decision Support for Multi-Attribute Problems with Even Swaps. <i>Debarun Bhattacharjya, Jeffrey Kephart</i>	72
Learning to Predict from Crowdsourced Data. <i>Wei Bi, Liwei Wang, James Kwok, Zhuowen Tu</i>	82
Lifted Tree-Reweighted Variational Inference. <i>Hung Bui, Tuyen Huynh, David Sontag</i>	92
Approximate Decentralized Bayesian Inference. <i>Trevor Campbell, Jonathan How</i>	102
Inferring latent structures via information inequalities. <i>Rafael Chaves, Lukas Luft, Thiago Maciel, David Gross, Dominik Janzing, Bernhard Schölkopf</i>	112
Near-optimal Adaptive Pool-based Active Learning with General Loss. <i>Nguyen Viet Cuong, Wee Sun Lee, Nan Ye</i>	122
A Permutation-Based Kernel Conditional Independence Test. <i>Gary Doran, Krikamol Muandet, Kun Zhang, Bernhard Schölkopf</i>	132
Parallel Markov Chain Monte Carlo for Pitman-Yor Mixture Models. <i>Kumar Dubey, Sinead Williamson, Eric Xing</i>	142

Market Making with Decreasing Utility for Information.	
<i>Miroslav Dudík, Rafael Frongillo, Jennifer Wortman Vaughan</i>	152
Universal Convexification via Risk-Aversion.	
<i>Krishnamurthy Dvijotham, Maryam Fazel, Emanuel Todorov</i>	162
Structured Proportional Jump Processes.	
<i>Tal El-Hay, Omer Weissbrod, Elad Eban, Maurizio Zazzi, Francesca Incardona</i>	172
Electing the Most Probable Without Eliminating the Irrational: Voting Over Intransitive Domains.	
<i>Edith Elkind, Nisarg Shah</i>	182
Iterative Splits of Quadratic Bounds for Scalable Binary Tensor Factorization.	
<i>Beyza Ermis, Guillaume Bouchard</i>	192
Finding Optimal Bayesian Network Structures with Constraints Learned from Data.	
<i>Xiannian Fan, Brandon Malone, Changhe Yuan</i>	200
Bisimulation Metrics are Optimal Value Functions.	
<i>Norm Ferns, Doina Precup</i>	210
Annealing Paths for the Evaluation of Topic Models.	
<i>James Foulds, Padhraic Smyth</i>	220
Active Learning of Linear Embeddings for Gaussian Processes.	
<i>Roman Garnett, Michael Osborne, Philipp Hennig</i>	230
Estimating Causal Effects by Bounding Confounding.	
<i>Philipp Geiger, Dominik Janzing, Bernhard Schölkopf</i>	240
Bayesian Optimization with Unknown Constraints.	
<i>Michael Gelbart, Jasper Snoek, Ryan Adams</i>	250
Nonparametric Clustering with Distance Dependent Hierarchies.	
<i>Soumya Ghosh, Michalis Raptis, Leonid Sigal, Erik Sudderth</i>	260
Transformation-based Probabilistic Clustering with Supervision.	
<i>Siddharth Gopal, Yiming Yang</i>	270
Understanding the Complexity of Lifted Inference and Asymmetric Weighted Model Counting.	
<i>Eric Gribkoff, Guy Van Den Broeck, Dan Suciu</i>	280
Interactive Learning from Unlabeled Instructions.	
<i>Jonathan Grizou, Iñaki Iturrate, Luis Montesano, Pierre-Yves Oudeyer, Manuel Lopes</i>	290
Batch-Mode Active Learning via Error Bound Minimization.	
<i>Quanquan Gu, Tong Zhang, Jiawei Han</i>	300
Efficient Bayesian Nonparametric Modelling of Structured Point Processes.	
<i>Tom Gunter, Chris Lloyd, Michael Osborne, Stephen Roberts</i>	310
Learning Peptide-Spectrum Alignment Models for Tandem Mass Spectrometry.	
<i>John Halloran, Jeffrey Bilmes, William Noble</i>	320
Off-policy TD(λ) with a true online equivalence.	
<i>Hado Van Hasselt, Rupam Mahmood, Rich Sutton</i>	330
Constraint-based Causal Discovery: Conflict Resolution with Answer Set Programming.	
<i>Antti Hyttinen, Frederick Eberhardt, Matti Järvisalo</i>	340
Generating structure of latent variable models for nested data.	
<i>Masakazu Ishihata, Tomoharu Iwata</i>	350
Monotone Closure of Relaxed Constraints in Submodular Optimization: Connections Between Minimization and Maximization.	
<i>Rishabh Iyer, Stefanie Jegelka, Jeffrey Bilmes</i>	360
Min- d -Occur: Ensuring Future Occurrences in Streaming Sets.	
<i>Vidit Jain, Sainyam Galhotra</i>	370
Instance Label Prediction by Dirichlet Process Multiple Instance Learning.	
<i>Melih Kandemir, Fred Hamprecht</i>	380
Closed-form Solutions to a Subclass of Continuous Stochastic Games via Symbolic Dynamic Programming.	
<i>Shamin Kinathil, Scott Sanner, Nicolás Della Penna</i>	390
Recursive Best-First AND/OR Search for Optimization in Graphical Models.	
<i>Akihiro Kishimoto, Radu Marinescu</i>	400

Saturated Conditional Independence with Fixed and Undetermined Sets of Incomplete Random Variables.	
<i>Henning Koehler, Sebastian Link</i>	410
Matroid Bandits: Fast Combinatorial Optimization with Learning.	
<i>Branislav Kveton, Zheng Wen, Azin Ashkan, Hoda Eydgahi, Brian Eriksson</i>	420
Multi-label Image Classification with A Probabilistic Label Enhancement Model.	
<i>Xin Li, Feipeng Zhao, Yuhong Guo</i>	430
Sequential Model-Based Ensemble Optimization.	
<i>Alexandre Lacoste, Hugo Larochelle, Mario Marchand, François Laviolette</i>	440
Position-Aware ListMLE: A Sequential Learning Process for Ranking.	
<i>Yanyan Lan, Yadong Zhu, Jiafeng Guo, Shuzi Niu, Xueqi Cheng</i>	449
Continuously indexed Potts models on unoriented graphs.	
<i>Loic Landrieu, Guillaume Obozinski</i>	459
Efficient Inference of Gaussian-Process-Modulated Renewal Processes with Application to Medical Event Data.	
<i>Thomas Lasko</i>	469
Optimal Resource Allocation with Semi-Bandit Feedback.	
<i>Tor Lattimore, Koby Crammer, Csaba Szepesvari</i>	477
Quantifying Nonlocal Informativeness in High-Dimensional, Loopy Gaussian Graphical Models.	
<i>Daniel Levine, Jonathan How</i>	487
CoRE Kernels.	
<i>Ping Li</i>	496
Efficient Sparse Recovery via Adaptive Non-Convex Regularizers with Oracle Property.	
<i>Ming Lin, Rong Jin, Changshui Zhang</i>	505
Nuclear Norm Regularized Least Squares Optimization on Grassmannian Manifolds.	
<i>Yuanyuan Liu, Fanhua Shang, Hong Cheng, James Cheng</i>	515
Fast Ridge Regression with Randomized Principal Component Analysis and Gradient Descent.	
<i>Yichao Lu, Dean Foster</i>	525
Adaptive Monotone Shrinkage for Regression.	
<i>Zhuang Ma, Dean Foster, Robert Stine</i>	533
Firefly Monte Carlo: Exact MCMC with Subsets of Data.	
<i>Dougal Maclaurin, Ryan Adams</i>	543
Sequential Bayesian Optimisation for Spatial-Temporal Monitoring.	
<i>Roman Marchant, Fabio Ramos, Scott Sanner</i>	553
AND/OR Search for Marginal MAP.	
<i>Radu Marinescu, Rina Dechter, Alexander Ihler</i>	563
Stochastic Discriminative EM.	
<i>Andres Masegosa</i>	573
Latent Kullback Leibler Control for Continuous-State Systems using Probabilistic Graphical Models.	
<i>Takamitsu Matsubara, Vicenç Gomez, Hilbert Kappen</i>	583
GPS-ABC: Gaussian Process Surrogate Approximate Bayesian Computation.	
<i>Edward Meeds, Max Welling</i>	593
Lifted Message Passing as Reparametrization of Graphical Models.	
<i>Martin Mladenov, Amir Globerson, Kristian Kersting</i>	603
Fast Gaussian Process Posteriors with Product Trees.	
<i>David Moore, Stuart Russell</i>	613
Asymptotically Exact, Embarrassingly Parallel MCMC.	
<i>Willie Neiswanger, Eric Xing, Chong Wang</i>	623
Modeling Citation Networks Using Latent Random Offsets.	
<i>Willie Neiswanger, Chong Wang, Qirong Ho, Eric Xing</i>	633
Collaborative Multi-output Gaussian Processes.	
<i>Trung Nguyen, Edwin Bonilla</i>	643
Combining predictions from linear models when training and test inputs differ.	
<i>Thijs Van Ommen</i>	653

Optimal amortized regret in every interval.	
<i>Rina Panigrahy, Preyas Popat</i>	663
Learning Partial Policies to Speedup MDP Tree Search.	
<i>Jervis Pinto, Alan Fern</i>	672
Estimating Accuracy from Unlabeled Data.	
<i>Emmanouil Antonios Platanios, Avrim Blum, Tom Mitchell</i>	682
k-NN Regression on Functional Data with Incomplete Observations.	
<i>Sashank J. Reddi, Barnabas Poczos</i>	692
Bayesian Inference in Treewidth-Bounded Graphical Models Without Indegree Constraints.	
<i>Daniel J. Rosenkrantz, Madhav V. Marathe, S. S. Ravi, Anil K. Vullikanti</i>	702
SPPM: Sparse Privacy Preserving Mappings.	
<i>Salman Salamatian, Nadia Fawaz, Branislav Kveton, Nina Taft</i>	712
Correlated Compressive Sensing for Networked Data.	
<i>Tianlin Shi, Da Tang, Liwen Xu, Thomas Moscibroda</i>	722
Improved Densification of One Permutation Hashing.	
<i>Anshumali Shrivastava, Ping Li</i>	732
First-Order Open-Universe POMDPs.	
<i>Siddharth Srivastava, Stuart Russell, Paul Ruan, Xiang Cheng</i>	742
A Hierarchical Switching Linear Dynamical System Applied to the Detection of Sepsis in Neonatal Condition Monitoring.	
<i>Ioan Stanculescu, Christopher K.I. Williams, Yvonne Freer</i>	752
A Spectral Algorithm for Learning Class-Based n -gram Models of Natural Language.	
<i>Karl Stratos, Do-kyum Kim, Michael Collins, Daniel Hsu</i>	762
Inference Complexity in Continuous Time Bayesian Networks.	
<i>Liessman Sturlaugson, John Sheppard</i>	772
Model Regularization for Stable Sample Rollouts.	
<i>Erik Talvitie</i>	780
HELM: Highly Efficient Learning of Mixed copula networks.	
<i>Yaniv Tenzer, Gal Elidan</i>	790
Metrics for Probabilistic Geometries.	
<i>Alessandra Tosi, Søren Hauberg, Alfredo Vellido, Neil Lawrence</i>	800
Efficient Regret Bounds for Online Bid Optimisation in Budget-Limited Sponsored Search Auctions.	
<i>Long Tran-Thanh, Lampros Stavrogiannis, Victor Naroditskiy, Valentin Robu, Nicholas Jennings, Peter Key</i>	809
A Consistent Estimator of the Expected Gradient Outerproduct.	
<i>Shubhendu Trivedi, Jialei Wang, Samory Kpotufe, Gregory Shakhnarovich</i>	819
Venn-Abers Predictors.	
<i>Vladimir Vovk, Ivan Petej</i>	829
Tightness Results for Local Consistency Relaxations in Continuous MRFs.	
<i>Yoav Wald, Amir Globerson</i>	839
Bayesian Filtering with Online Gaussian Process Latent Variable Models.	
<i>Yali Wang, Marcus Brubaker, Brahim Chaib-draa, Raquel Urtasun</i>	849
Approximating the Bethe Partition Function.	
<i>Adrian Weller, Tony Jebara</i>	858
Understanding the Bethe Approximation: When and How can it go Wrong?.	
<i>Adrian Weller, Kui Tang, Tony Jebara, David Sontag</i>	868
A Bayesian Nonparametric Model for Spectral Estimation of Metastable Systems.	
<i>Hao Wu</i>	878
Fast Newton methods for the group fused lasso.	
<i>Matt Wytock, Suvrit Sra, J. Zico Kolter</i>	888
Learning from Point Sets with Observational Bias.	
<i>Liang Xiong, Jeff Schneider</i>	898
Constructing Separators and Adjustment Sets in Ancestral Graphs.	
<i>Benito van der Zander, Maciej Liskiewicz, Johannes Textor</i>	907

Belief-Kinematics Jeffrey's Rules in the Theory of Evidence.	
<i>Chunlai Zhou, Mingyue Wang, Biao Qin</i>	917

Preface

The Conference on Uncertainty in Artificial Intelligence (UAI) is the premier international conference on research related to representation, inference, learning and decision making in the presence of uncertainty within the field of Artificial Intelligence. This volume contains all papers that were accepted for the 30th UAI Conference, held in Quebec City, Quebec, Canada, from July 23rd to 27th 2014. Papers appearing in this volume were subjected to a rigorous review process. 292 papers were submitted to the conference and each was peer-reviewed by 3 or more reviewers with the supervision by one Senior Program Committee member. A total of 94 papers were accepted, 24 for oral presentation and 70 for poster presentation, for an acceptance rate of 32%. We are very grateful to the program committee and senior program committee members for their diligent efforts. We are confident that the proceedings, like past UAI conference proceedings, will become an important archival reference for the field.

We are pleased to announce that the Microsoft Best Paper Award was given to Dougal Maclaurin and Ryan Adams for their paper “Firefly Monte Carlo: Exact MCMC with Subsets of Data”. The IBM Best Student Paper Award was given to Benito van der Zander (co-authored with Maciej Liskiewicz and Johannes Textor) for their paper “Constructing Separators and Adjustment Sets in Ancestral Graphs”. The Google Best Student Paper Award was given to Nguyen Viet Cuong (co-authored with Wee Sun Lee and Nan Ye) for their paper “Near-optimal Adaptive Pool-based Active Learning with General Loss”. The Facebook Best Student Paper Award was given to Krishnamurthy Dvijotham (co-authored with Maryam Fazel and Emanuel Todorov) for their paper “Universal Convexification via Risk-Aversion”. And the Best Paper Runner-Up was “Optimal Resource Allocation with Semi-Bandit Feedback” by Tor Lattimore, Koby Crammer, and Csaba Szepesvari.

In addition to the presentation of technical papers, we were very pleased to have five distinguished invited speakers at UAI 2014: David M. Blei (Columbia University), Craig Boutilier (University of Toronto), Michael L. Littman (Brown University), Andrew Ng (Stanford University), and, as Banquet Speaker, Yann LeCun (Facebook and NYU). Another interesting addition to the conference program was the Fifth UAI Probabilistic Inference Competition, organized by Vibhav Gogate.

The UAI 2014 tutorials program, chaired by Vibhav Gogate, consisted of four tutorials: “Random Perturbations for Inference” by Tamir Hazan, “Learning Tractable Probabilistic Models” by Pedro Domingos and Daniel Lowd, “Probabilistic Programming” by Avi Pfeffer, and “Probabilistic Inference in Relational Models” by Dan Suciu and Guy Van den Broeck.

UAI 2014 also hosted two one-day workshops (organized by workshops chair John Mark Agosta): “11th Bayesian Applications Workshop” and “Causal Inference: Learning and Prediction”.

Jin Tian and Nevin L. Zhang (Program Co-Chairs)
Ann Nicholson (General Chair)

Organizing Committee

General Chair

Ann Nicholson, Monash University/Bayesian Intelligence, Australia

Program Chairs

Nevin L. Zhang, The Hong Kong University of Science and Technology, China

Jin Tian, Iowa State University, USA

Proceedings Chair

Daniel Lowd, University of Oregon, USA

Tutorials Chair

Vibhav Gogate, University of Texas at Dallas, USA

Workshops Chair

John Mark Agosta, C9 Inc., San Mateo, CA, USA

Publicity Chair

Changhe Yuan, City University of New York, USA

Acknowledgments

The success of a conference such as UAI depends greatly on the efforts of many individuals who volunteer their time to provide expert and detailed reviews of submitted papers. In particular, the Program Committee and Senior Program Committee for UAI 2014 were responsible for generating reviews and recommendations for the 292 submissions to the conference. Each submitted paper was reviewed by at least 3 members of the Program Committee. The Senior Program Committee then assessed the individual reviews for each paper, moderated discussion among Program Committee members if needed, and generated meta-reviews and recommendations for the program chairs. We are extremely grateful for the efforts of all of the individuals listed below.

Senior Program Committee

Jeff Bilmes	University of Washington
Craig Boutilier	University of Toronto
Joaquin Candela	Facebook
Max Chickering	Microsoft Research
Fabio Cuzzolin	Oxford Brookes University
Adnan Darwiche	UCLA
Denver Dash	Intel Labs Pittsburgh
Cassio de Campos	IDSIA
Rina Dechter	UC-Irvine
Francisco Diez	UNED
Gal Elidan	The Hebrew University of Jerusalem
Tom Heskes	Radboud University Nijmegen
Eric Horvitz	Microsoft Research
Alexander Ihler	UC Irvine
Tommy Jaakkola	MIT
Dominik Janzing	Max Planck Institute
Helge Langseth	The Norwegian University of Science and Technology
Kathryn Laskey	George Mason University
Tze-Yun Leong	National University of Singapore
Chris Meek	Microsoft Research
Remi Munos	INRIA Lille
Petri Myllymaki	Helsinki Institute for Information Technology
Thomas Nielsen	Aalborg University
Peter Grünwald	Centrum voor Wiskunde en Informatica
David Poole	University of British Columbia
Thomas Richardson	University of Washington
Dale Schuurmans	University of Alberta
Prakash Shenoy	University of Kansas
Ricardo Silva	University College London
David Sontag	New York University
Peter Spirtes	Carnegie Mellon University
Yi Wang	A*STAR Singapore
Dit-Yan Yeung	Hong Kong University of Science and Technology
Jun Zhu	Tsinghua University
Shlomo Zilberstein	University of Massachusetts Amherst

Program Committee

John Mark Agosta	C9 Inc.
Ayesha Ali	University of Guelph
Russell Almond	Florida State University
Christopher Amato	MIT
Leila Amgoud	IRIT - Universite Paul Sabatier
Eyal Amir	University of Illinois at Urbana-Champaign
Animashree Anandkumar	UC Irvine
Alessandro Antonucci	IDSIA
Cedric Archambeau	Amazon Berlin
Nimar Arora	Oracle
Elias Bareinboim	UCLA
Dhruv Batra	Virginia Tech
Carlo Berzuni	University of Manchester
Debarun Bhattacharjya	IBM Research
Bozhena Bidyuk	Google
Antoine Bordes	CNRS
Guillaume Bouchard	Xerox Research Centre Europe
Alexandre Bouchard-Cote	UBC
Ronen Brafman	Ben-Gurion University
Darius Brazunas	University of Toronto
Marcus Brubaker	Toyota Technological Institute at Chicago
Emma Brunskill	Carnegie Mellon University
Olivier Buffet	LORIA-INRIA, France
Wray Buntine	NICTA's Canberra Lab
Cory Butz	University of Regina, Canada
Erik Cambria	National University of Singapore
Lawrence Carin	Duke
Robert Castelo	Universitat Pompeu Fabra
Laurent Charlin	University of Toronto
Kamalika Chaudhuri	UC San Diego
Tao Chen	Renren games
Arthur Choi	UCLA
Tianjiao Chu	University of Pittsburgh
Tom Claassen	Radboud University Nijmegen
Ira Cohen	Hewlett Packard Labs
Mark Crowley	Oregon State University, USA
James Cussens	University of York
Sebastien Destercke	CNRS
Marek Druzdzal	University of Pittsburgh
David Dunson	Duke University
Jennifer Dy	Northeastern University
Frederick Eberhardt	Caltech
Robin Evans	University of Oxford
Helene Fargier	IRIT-CNRS
Alan Fern	Oregon State University
M. Julia Flores	University of Castilla - La Mancha (UCLM)
Aram Galstyan	Information Sciences Institute
Kuzman Ganchev	Google Research
Minos Garafalakis	Technical University of Crete
Roman Garnett	University of Bonn
Phan Giang	George Mason University
Mark Girolami	University College London
Bob Givan	Purdue University ECE

Amir Globerson	Hebrew University
Lluís Godó	Artificial Intelligence Research Institute
Vibhav Gogate	University of Texas at Dallas
Vicenc Gómez	Radboud University, Nijmegen
Manuel Gómez-Olmedo	Universidad de Granada
Christophe Gonzales	LIP6-UPMC
Noah Goodman	Stanford University
Mihajlo Grbovic	Yahoo! Labs
Moritz Grosse-Wentrup	Max Planck Institute for Intelligent Systems
Amit Gruber	Yahoo!
Yuhong Guo	Temple University
Maya Gupta	University of Washington
Philipp Hennig	Max Planck Institute
Jesse Hoey	University of Waterloo
Arjen Hommersom	University of Nijmegen
Antti Honkela	University of Helsinki
William Hsu	Kansas State University
Bert Huang	University of Maryland
Marcus Hutter	Australian National University
Antti Hyttinen	California Institute of Technology
David Jensen	University of Massachusetts Amherst
Nebojsa Jojic	Microsoft Research
Kristian Kersting	Fraunhofer IAIS
David Knowles	Stanford University
Mikko Koivisto	Helsinki Institute for Information Technology
Kevin Korb	Monash University
Wojciech Kotłowski	Poznań University of Technology
Akshat Kumar	IBM Research India
Branislav Kveton	Technicolor Labs
Jerome Lang	LAMSADE, CNRS, & Université Paris-Dauphine
Su-In Lee	University of Washington
Jan Lemeire	Vrije Universiteit Brussel
Philippe Leray	University of Nantes
Lei Li	Florida International University
Wu-Jun Li	Nanjing University
Michael Littman	Brown University
Qiang Liu	UC Irvine
Weiru Liu	Queen's University Belfast
Ying Liu	MIT
Dan Lizotte	University of Waterloo
Daniel Lowd	University of Oregon
Aurelie Lozano	IBM Research
Michael Lyu	Chinese University of Hong Kong
Marloes Maathuis	ETH Zurich
Malik Magdon-Ismaïl	Rensselaer Polytechnic Institute
Brandon Malone	Helsinki Institute for Information Technology, Finland
Vikash Mansinghka	MIT
Radu Marinescu	IBM Research
Maria Vanina Martinez	University of Oxford
Ole Mengshoel	Carnegie Mellon University
Ofer Meshi	Toyota Technological Institute at Chicago
Taneli Mielikainen	Nokia Research Center Palo Alto
Brian Milch	Google
David Mimno	Princeton
Thomas Minka	Microsoft Research UK

Andriy Mnih	Gatsby/University College London
Claire Monteleoni	George Washington University
Joris Mooij	University of Amsterdam
Iain Murray	University of Edinburgh
Sriraam Natarajan	Indiana University
Mathias Niepert	University of Washington
William Noble	University of Washington
Nuria Oliver	Telefonica
Michael Osborne	Oxford University
David Page	UW Madison
John Paisley	Columbia University
Hector Palacios	Pompeu Fabra University
Jose Pena	Linkoping University
David Pennock	Microsoft Research
Jonas Peters	ETH Zurich
Marek Petrik	IBM Research
Kim-Leng Poh	National University of Singapore
Leonard Poon	Hong Kong Institute of Education
Pascal Poupart	University of Waterloo
Bob Price	PARC
David Pynadath	USC Institute for Creative Technologies
Yuan (Alan) Qi	Purdue University
Erik Quaeghebeur	Centrum Wiskunde & Informatica
Piyush Rai	Duke University
Roland Ramsahai	University of Cambridge
Mark Reid	Australian National University
Silja Renooij	Univesiteit Utrecht
Teemu Roos	Helsinki Institute for Information Technology
Rafael Rumi	Almeria University
Brian Ruttenberg	Charles River Analytics
Regis Sabbadin	INRA
Antonio Salmeron	Universidad de Almeria
Scott Sanner	NICTA and the Australian National University
Bart Selman	Department of Computer Science
Gerardo Simari	Oxford
Tomas Singliar	Amazon
Mathieu Sinn	IBM Research - Ireland
Le Song	Georgia Tech
L. Enrique Sucar	INAOE, Mexico
Joe Suzuki	Osaka University
Umar Syed	Google
Vincent Tan	National University of Singapore, Singapore
Pingzhong Tang	Tsinghua University
Danny Tarlow	Microsoft Research
Graham Taylor	University of New York
Florent Teichteil-Konigsbuch	ONERA - The French Aerospace Lab
Matthias Troffaes	University of Durham
Ioannis Tsamardinos	University of Crete
Raquel Urtasun	Toyota Technological Institute at Chicago
Marco Valtorta	University of South Carolina
Laurens van der Maaten	Delft University of Technology
Greg Ver Steeg	Information Sciences Institute
S V N Vishwanathan	Purdue University
Jirka Vomlel	Institute of Information Theory and Automation, Czech Republic
Yevgeniy Vorobeychik	Vanderbilt University

Vladimir Vovk	Royal Holloway
Thomas Walsh	MIT
Chong Wang	Carnegie Mellon University
Hao Wang	University of South Carolina
Paul Weng	Paris 6 University
Sinead Williamson	University of Texas at Austin
David Wipf	Microsoft Research Asia
Stefan Witwicki	Ecole Polytechnique Federale de Lausanne
Yang Xiang	University of Guelph, Canada
Changhe Yuan	City University of New York
Xiaotong Yuan	Nanjing University of Information Science and Technology
Chongjie Zhang	University of Massachusetts Amherst
Jiji Zhang	Lingnan University
Kun Zhang	MPI for Intelligent Systems
Yu Zhang	Hong Kong Baptist University
Onno Zoeter	Xerox Research Centre Europe

Additional Reviewers

Ryan Adams, Harvard	Vaclav Lin, UTIA, Academy of Sciences of the Czech Republic
Faruk Ahmed, Virginia Tech	Guang Ling, The Chinese University of Hong Kong
Udi Apsel, Ben Gurion University, Israel	Jie Liu, University of Wisconsin-Madison
Azin Ashkan, Technicolor Labs	Qi Lou, Virginia Tech
Prakriti Banik, Virginia Tech	Dougal Maclaurin, Harvard
Kim Bauters, Queen's University Belfast	Juan Mancilla, University of Illinois, Urbana-Champaign
Alessio Benavoli, Dalle Molle Institute for Artificial Intelligence	Ana M. Martínez, Monash University, Australia
Giorgos Borboudakis, University of Crete	Steffen Michels, University of Nijmegen
Robert Busa-Fekete, University of Paris	Umut Oztok, UCLA
Alberto Giovanni Busetto, ETH Zurich	Pere Pardo, University of Sevilla, Spain
KC Chang, George Mason University	Michael Perlman, University of Washington
Shouyuan Chen, The Chinese University of Hong Kong	Dongzhen Piao, CMU
Suming J. Chen, UCLA	Wen Pu, LinkedIn, University of Illinois Urbana-Champaign
Chen Cheng, The Chinese University of Hong Kong	José M. Puerta, UCLM, Spain
Mayank Daswani, Australian National University	Parameshwaran Raman, Purdue University
Michael Davis, Queen's University Belfast	Clint Solomon, Virginia Tech
Ariel Deagustini, Universidad Nacional del Sur Bahia Blanca	Qing Sun, Virginia Tech
Nemanja Djuric, Yahoo Labs	Priya Sundararajan, CMU
Efrat Egozi-Levi, HP	Peter Sunehag, Australian National University
Doris Entner, V-Research, Industrial R&D	Paolo Viappiani, LIP6-UPMC
Hugo Jair Escalante, INAOE, Mexico	Shenlong Wang, University of Toronto
José A. Gámez, UCLM, Spain	Zheng Wen, Stanford University
Codruta Girlea, University of Illinois Urbana-Champaign	Wim Wiegierinck, Radboud University Nijmegen
Marek Grzes, University of Waterloo	Jiasen Yang, Purdue University
Rajesh Kalyanam, Purdue University	Shenglin Zhao, The Chinese University of Hong Kong
Vincenzo Lagani, Institute of Computer Science, Foundation for Research and Technology, Hellas	Jieming Zhu, The Chinese University of Hong Kong
Jan Leike, Australian National University	

Additional Acknowledgments

A number of other people have made significant contributions towards making UAI 2014 possible. We acknowledge and thank:

- Fabio Cozman, Nando de Freitas, Peter Grunwald, David Poole, Prakash Shenoy and Padhraic Smyth for general advice and assistance related to UAI organization.
- Rina Dechter and Gal Elidan for initiating the inference competition.
- Thao Hoang for creating and maintaining the UAI 2014 Web site.
- Ewelina Akehurst for providing assistance to the general chair
- Karen Wong for assistance in setting up UAI 2014 Website.
- Kilian Weinberger for setting up the automatic paper formatting checker.

Sponsors

We gratefully acknowledge the generous support provided by our sponsors, including support for student best paper awards. Without our sponsors' support it would not be feasible to organize a conference such as UAI 2014 without charging much higher registration fees.

Microsoft®
Research



facebook®



charles river analytics

IBM Research

Best Paper Awards

Best Paper Award - sponsored by Microsoft

Firefly Monte Carlo: Exact MCMC with Subsets of Data

Dougal Maclaurin, Ryan Adams

Best Paper Runner-Up

Optimal Resource Allocation with Semi-Bandit Feedback

Tor Lattimore, Koby Crammer, Csaba Szepesvari

IBM Best Student Paper

Constructing Separators and Adjustment Sets in Ancestral Graphs

Benito van der Zander, Maciej Liskiewicz, Johannes Textor

Google Best Student Paper

Near-optimal Adaptive Pool-based Active Learning with General Loss

Nguyen Viet Cuong, Wee Sun Lee, Nan Ye

Facebook Best Student Paper

Universal Convexification via Risk-Aversion

Krishnamurthy Dvijotham, Maryam Fazel, Emanuel Todorov

Proceedings

MEMR: A Margin Equipped Monotone Retargeting Framework for Ranking

Sreangsu Acharyya *

Dept. of Electrical Engineering
University of Texas Austin.

Joydeep Ghosh

Dept. of Electrical Engineering
University of Texas Austin.

Abstract

We bring to bear the tools of convexity, margins and the newly proposed technique of monotone retargeting upon the task of learning permutations from examples. This leads to novel and efficient algorithms with guaranteed prediction performance in the online setting and on global optimality and the rate of convergence in the batch setting. Monotone retargeting efficiently optimizes over all possible monotone transformations as well as the finite dimensional parameters of the model. As a result we obtain an effective algorithm to learn transitive relationships over items. It captures the inherent combinatorial characteristics of the output space yet it has a computational burden not much more than that of a generalized linear model.

1 INTRODUCTION

Many applications require items to be ordered correctly. Prototypical examples of such applications are information retrieval and recommender systems. In most cases, however, the quality measure that actually defines the *transitive relation* of interest can be accessed only through examples. This lack of direct access to the ordering relation motivates learning the quality measure from the covariates of the items. We distinguish this task from a related and easier one of learning binary pairwise relations where transitivity is not required by the application.

Existing techniques of learning to rank (LETOR) fall under 3 categories: (i) point-wise methods, (ii) pair-wise methods and (iii) list-wise methods. In point-wise methods, higher ranked items are assigned higher target scores. The method ignores the combinatorial

structure of the output space and regresses the scores directly. Pair-wise methods capture some structure by trying to classify for a pair whether the first item in the pair out-ranks the second. Their predictions need not be transitive and an *order-reconciliation step* is necessary to enforce it. This is NP hard [8], necessitating approximations and heuristics. Finally, there are list-wise methods that model the full combinatorial structure and need to solve formidable optimization problems. They have to cut corners for scalability. Notable approaches include sampling [25], approximations [2], and resorting to point-wise methods [6].

An ideal LETOR formulation should (i) capture combinatorial structure like list-wise methods, but with (ii) algorithms as simple as point-wise methods. While this seems too much to ask, the recently proposed monotone retargeting (MR) technique is one way how this may be approached [1]. MR outperforms several state of the art ranking algorithms such as Listnet [6] and RankCosine, even after improving those algorithms for statistical consistency as proposed by Ravikumar et. al. [21].

MR efficiently reduces, the LETOR problem to a generalized linear model (GLM) with no loss in generality. It subsumes *statistically consistent* methods of [21]. The distinguishing characteristic of MR is its “*re-targeting*” *paradigm*, where instead of fitting training scores exactly, it tries to fit any score that captures the desired order. Recall that our task is to retrieve the correct order and not the training scores. In this setting, retrieving the specified training scores are an unnecessary burden. The specified training scores may be particularly difficult to fit for the chosen family of regression function class, but there might exist score assignments that capture the desired order and also simultaneously lie in the range space of the regression function class being used. The MR framework tries to find such score assignments by formulating it as a Bregman divergence minimization problem.

In this paper we push the *retargeting* idea further.

*Authors acknowledge NSF grant IIS-1017614

This is facilitated by (i) a remarkably efficient finite time optimization over the infinite space of all monotonic transformations and (ii) properties of Bregman divergences particularly suited for learning orders.

Let us draw a few analogies from classification. A pointwise approach to a $\{-1, 1\}$ encoded classification problem would try to fit the $\{-1, 1\}$ training scores exactly, possibly enriching the approximating function class till the quality of the fit is acceptable. Most successful classifiers, however, fit values that are discriminable, ignoring, entirely, whether they are close to the training scores of $\{-1, 1\}$ in value.

The MR cost function consists of two parts: a loss and a regularization. Similar to perceptrons, the moment MR predictions retrieve the training ranks, its loss drops to zero. Experience in classification has taught us that losses that continue to be active after training error has dropped to zero yield better accuracy, for example, SVMs, logistic regression and boosting. In our paper we equip MR with such a margin-like property. This can be done in a few different ways. Our intent is not to champion one over another. This paper is not about advocacy, but about exploring how margin may be incorporated into the “*retargeting*” paradigm.

In this paper (i) we introduce large and fixed margin variants of the MR approach. Without margins the MR cost function is degenerate, an aspect that is not developed in the previous work [1]. Unlike the previous approach, we model the requirement of a margin explicitly in this paper. (ii) Unlike [1] we are able to model the notion that ordering errors at the top are worse than those at the bottom. (iii) It was shown that MR cost function is jointly convex *iff* the Bregman divergence chosen is squared Euclidean. We extend the formulation to enable joint convexity to all strongly convex Bregman divergence, not to advocate non-Euclidean divergences but to explore them.

Joint convexity has two important ramifications: one affects ease of evaluation of the technique, the other affects efficiency of training. The initialization independence of the optimum, gained as a result of convexity induced uniqueness, makes comparing different Bregman divergences easier, eliminating the need for multiple initializations during training. (iv) On the other hand for training, joint convexity allows us to replace *exact* coordinate-wise updates that were used in [1] with more efficient gradient updates with guarantees on global optimality. (v) This yields efficient online algorithms with regret bounds over permutations. Finally, (vi) we provide rates of convergence guarantees, an aspect missing from the previous work.

To date many cost functions have been designed to *evaluate* rankings, for example, discounted cumula-

tive gain (DCG), normalized discounted cumulative gain (NDCG) [13], expected reciprocal rank (ERR) [7], mean average precision (MAP) [3]. They are functions of permutations and capture the notion that positional accuracy at the top is more important than at the bottom. They are reasonably easy to compute given a ranking, but to optimize them in training is notoriously intractable. Our formulation, on the other hand, introduces a family of cost functions that have characteristics desired in ranking: dependence on order not on scores and the ability to capture the importance of non-uniform positional accuracy, but at the same time optimized *globally* with ease. These aspects set our work apart from other approaches of learning to rank.

We follow the **notation** used in the MR paper. Vectors are denoted by bold lower case letters, matrices are capitalized. \mathbf{x}^\dagger is \mathbf{x} transposed and $\|\mathbf{x}\|$ its L_2 norm. $\text{Adj-Diff}(\cdot)$ is the adjacent difference operator, and $\text{Cum-Sum}(\text{Adj-Diff}(\mathbf{x})) = \mathbf{x}$. \mathbf{x} is in *descending order* if $x_i \geq x_j$ when $i > j$. the set of such vectors is \mathcal{R}_\downarrow . \mathbf{x} is isotonic with \mathbf{y} if $x_i \geq x_j$ implies $y_i \geq y_j$. Δ denotes an unit simplex and Δ_ϵ its subset with members component-wise bounded away from 0 by ϵ . \mathbb{R}_+^d is the positive orthant and R_ϵ^d its subset similarly bounded away from 0 by ϵ . Interior is denoted by int .

2 BACKGROUND

We will use **Bregman Divergences** to construct our cost function. Let $\phi : \Theta \mapsto \mathbb{R}$, $\Theta = \text{dom } \phi \subseteq \mathbb{R}^d$ be a strictly convex, closed function, differentiable on $\text{int } \Theta$. The corresponding Bregman divergence $D_\phi(\cdot \| \cdot) : \text{dom}(\phi) \times \text{int}(\text{dom}(\phi)) \mapsto \mathbb{R}_+$ is defined as $D_\phi(\mathbf{x} \| \mathbf{y}) \triangleq \phi(\mathbf{x}) - \phi(\mathbf{y}) - \langle \mathbf{x} - \mathbf{y}, \nabla \phi(\mathbf{y}) \rangle$. From strict convexity it follows that $D_\phi(\mathbf{x} \| \mathbf{y}) \geq 0$ and $D_\phi(\mathbf{x} \| \mathbf{y}) = 0$ iff $\mathbf{x} = \mathbf{y}$. Bregman divergences are (strictly) convex in their first argument, but not necessarily convex in their second.

In this paper we only consider functions $\phi(\cdot) : \mathbb{R}^n \ni \mathbf{x} \mapsto \sum_i w_i \phi(x_i)$ that are weighted sums of *identical* scalar convex functions applied to each component, the former referred to as *weighted, identically separable (WIS)* or **IS** if the weights are equal. [1] and [21] identify this class to have properties particularly suited for ranking. The MR approach, in concert with Bregman divergences can provide compelling guarantees that includes convergence, parallelizability, statistical consistency, and avoids solving a linear assignment problem in every iteration of their training loop. Many LETOR algorithms [24], [25] fall prey to the latter.

Monotone Retargeting: The ranking problem in-

volves set of queries $\mathcal{Q} = \{q_1, q_i \dots q_{|\mathcal{Q}|}\}$ and a set of training items \mathcal{V} . For every query q_i , the elements of $\mathcal{V}_i \subset \mathcal{V}$ are ordered based on their relevance to the query. This ordering is expressed through a rank score vector $\tilde{\mathbf{r}}_i \in \mathbb{R}^{|\mathcal{V}_i|}$ whose components \tilde{r}_{ij} correspond to items in \mathcal{V}_i . Beyond establishing the order, the actual values are irrelevant. In our formulation, however, one may choose whether to treat these as irrelevant or incorporate them in the *retargeting* step, making the formulation more flexible.

For a query q_i the index j of \tilde{r}_{ij} is local to \mathcal{V}_i and assigned such that \tilde{r}_{ij} are in descending order for any \mathcal{V}_i . For every pair $\{q_i, v_{ij}\}$ a feature vector $\mathbb{R}^n \ni \mathbf{a}_{ij} = F(q_i, v_{ij})$ is an input to the algorithm, \mathbf{A}_i is a matrix whose j^{th} row is \mathbf{a}_{ij}^\dagger . The following formulation seems suitable for ranking:

$$\min_{\mathbf{w}, \Upsilon_i \in \mathcal{M}} \sum_i D_i(\tilde{\mathbf{r}}_i, \Upsilon_i \circ f(\mathbf{A}_i, \mathbf{w})), \quad (1)$$

where $D_i : \mathbb{R}^{|\mathcal{V}_i|} \times \mathbb{R}^{|\mathcal{V}_i|} \mapsto \mathbb{R}_+$ is some distance-like loss function, $f : \mathbb{R}^{|\mathcal{V}_i| \times n} \times \mathbb{R}^n \mapsto \mathbb{R}^{|\mathcal{V}_i|}$ is some parametric form with the parameter \mathbf{w} and $\Upsilon_i : \mathbb{R}^{|\mathcal{V}_i|} \mapsto \mathbb{R}^{|\mathcal{V}_i|}$ is a mapping that transforms the components by a scalar, strictly monotonic increasing function Υ_i , and \mathcal{M} is the class of all such functions. Formulation (1) avoids the problem that adversely affects point-wise-methods: solving an unnecessarily hard problem of matching the scores by value.

To avoid working in the space of \mathcal{M} which is infinite dimensional, MR solves a qualitative equivalent

$$\min_{\mathbf{w}, \mathbf{r} \in \mathcal{R}_{\downarrow_i}} \sum_i D_i(\mathbf{r}_i, f(\mathbf{A}_i, \mathbf{w})) \text{ s.t. } \mathcal{R}_{\downarrow_i} = \{\mathbf{r} | \exists \mathbf{M} \in \mathcal{M} \text{ s.t. } \mathbf{M}(\tilde{\mathbf{r}}_i) = \mathbf{r}\}. \quad (2)$$

Let us take a closer look at the constraint set used in formulation (2): Instead of considering all strictly increasing monotonic transforms Υ_i of the right argument, MR considers all inverse monotonic transformations of the left argument. This, remarkably, is a finite dimensional optimization problem because $\mathcal{R}_{\downarrow_i}$, the set of all vectors isotonic with $\tilde{\mathbf{r}}_i$ is a finitely characterizable convex cone. Motivated by convexity, MR chooses $D_i(\cdot, \cdot)$ to be a Bregman divergence $D_\phi(\cdot || \cdot)$ and $f(\mathbf{A}_i, \mathbf{w})$ to be $(\nabla \phi)^{-1}(\mathbf{A}_i \mathbf{w})$ to obtain¹

$$\min_{\beta_i, \mathbf{w}, \mathbf{r}_i \in \mathcal{R}_{\downarrow_i} \cap \mathcal{S}_i} \sum_{i=1}^{|\mathcal{Q}|} \frac{1}{|\mathcal{V}_i|} D_\phi(\mathbf{r}_i || (\nabla \phi)^{-1}(\mathbf{A}_i \mathbf{w} + \beta_i \mathbf{1})) + \frac{C}{2} \|\mathbf{w}\|^2. \quad (3)$$

¹We take a shortcut of writing $D_\phi(\cdot || (\nabla \phi)^{-1}(\cdot))$ instead of $D_{\phi_i}(\cdot, (\nabla \phi)^{-1})$ where ϕ_i indicates a separable convex function of an input dimension d_i built from component-wise sum of scalar function $\phi(\cdot)$.

$$\begin{aligned} \mathbf{P}_i^{t+1} &= \underset{\pi}{\text{Argmin}} D_\phi(\mathbf{r}_i^t || (\nabla \phi)^{-1}(\pi \mathbf{A}_i \mathbf{w}^t + \beta_i^t)) \quad \forall i \\ & \quad (4) \\ \mathbf{r}_i^{t+1} &= \underset{\mathbf{r} \in \mathcal{R}_{\downarrow_i} \cap \mathcal{S}_i}{\text{Argmin}} D_\phi(\mathbf{r} || (\nabla \phi)^{-1}(\mathbf{P}_i^{t+1} \mathbf{A}_i \mathbf{w}^t + \beta_i^t)) \quad \forall i \\ & \quad (5) \\ \mathbf{w}^{t+1}, \{\beta_i^{t+1}\} &= \\ & \quad \underset{\mathbf{w}, \{\beta_i\}}{\text{Argmin}} \sum_{i=1}^{|\mathcal{Q}|} D_\phi(\mathbf{r}_i^{t+1} || (\nabla \phi)^{-1}(\mathbf{P}_i^{t+1} \mathbf{A}_i \mathbf{w} + \beta_i^t)) \frac{C}{2} \|\mathbf{w}\|^2 \\ & \quad (7) \end{aligned}$$

Figure 1: Updates of Monotone Retargeting

where $(\nabla \phi)^{-1}$ is the inverse of the gradient mapping, \mathcal{S}_i is a convenient convex set excluding $\mathbf{0}$, that is necessary only for technical reasons.

In practice, even if \mathcal{V}_i is totally ordered, it is common to have a part of that information erased by quantization in the scores $\tilde{\mathbf{r}}$. MR deals with this by optimizing over block diagonal permutation matrices \mathbf{P}_i that permute contiguous blocks of indices that correspond to items whose relative order have been erased. The model is trained by iterating over the updates (4), (5) and (7) shown in Figure 1. It has been shown that these *exact coordinate-wise minimizations* updates converge to a local minimum (or *global for square loss* [1]) of function (3). Update (4) is accomplished by sorting. This turns out to be so because of special properties of separable Bregman divergences (see [1] for details). Update (5) uses the exponentiated gradient algorithm [15] and (7) is the same problem as estimating the parameters of a generalized linear model [19]. A quasi-Newton method (LBFGS [17]) was used to solve (7). In the rest of the paper the block diagonal permutation matrices \mathbf{P}_i will be suppressed. Our extensions continue to be effective for partial order via updates that correspond to (4), but this is not elaborated further for brevity.

3 FORMULATION

The rest of the paper describes our contribution. Its prominent features are: (i) formulation of fixed and large margin aspects, (ii) *joint* convexity of the cost function in the targets \mathbf{r} and the parameters \mathbf{w} , which yields (iii) guarantees on performance in the online setting and super-linear convergence in the batch setting.

Since there are multiple moving parts in our formulation, it is easy to get lost in the details. To preempt that we lay out the flow of our arguments. We explain the formulation by modifying the cost function (3) suc-

cessively. We conclude each subsection with summary of what has been achieved in the subsection so far.

Convexity: We equip the cost function with strong and joint convexity, aspects missing in the original work. We pick a *matching* form of the regularizer so that it adds no extra computational burden and quantify the amount of regularization that is sufficient to guarantee joint convexity. It may not be surprising that regularization extends convexity properties to MR losses other than squared Euclidean. What is surprising, however, is that this convexity applies jointly to \mathbf{r} and \mathbf{w} although the regularizers themselves are separated. Strong joint convexity and smoothness thus gained lead to the performance and convergence guarantees. This is the topic of section 3.1.

Margins: Second we plug a loophole in the MR cost function by ensuring margins between all adjacent target scores $r_{i,j}, r_{i,j+1}$. Without this, the cost function (3) is degenerate: one can achieve zero loss by setting $\mathbf{w}, \beta = \mathbf{0}$. We provide different ways of ensuring this: (i) directly by setting constraints, and (ii) indirectly by rewarding margins. Since both the constraints and the rewards are linear, this does not disrupt joint convexity. The key is to optimize the modified cost function efficiently. This is the topic of section 3.3.

3.1 Convexity, Smoothness and Optimization

MR ensures joint convexity *only* if squared Euclidean distance is used. We incorporate joint convexity into the cost function (3). This benefits us in two ways: (i) it removes initialization dependence of the training method and (ii) as we shall see, allows for a more efficient method of training, both online and batch with excellent convergence rates. We know that strong convexity together with smooth gradients (and Hessians for second order methods) admit efficient minimization: gradient descent achieves linear rate of convergence, quasi-Newton (truncated-Newton) achieves superlinear rates. We examine conditions under which our ranking formulations have these properties.

3.1.1 Joint Convexity

Let $\phi(\cdot)$ be s strongly convex [5]. Consider the term:

$$F_i(\mathbf{r}_i, \mathbf{w}) = \frac{1}{|\mathcal{V}_i|} \left(D_\phi(\mathbf{r}_i) \left\| (\nabla \phi)^{-1}(\mathbf{A}_i \mathbf{w}) \right\|^2 + \underbrace{C_{r_i} D_\phi(\mathbf{r}_i) \left\| \mathbf{q}_i \right\|^2 + \frac{C_{w_i}}{2} \|\mathbf{w}\|_{\mathbf{A}_i}^2}_{\text{Regularization terms}} \right) \quad (8)$$

using which we modify cost function (3) to

$$F(\{\mathbf{r}_i\}, \mathbf{w}) = \sum_i \frac{|\mathcal{Q}|}{|\mathcal{V}_i|} F_i(\mathbf{r}_i, \mathbf{w}) + \frac{C}{2} \|\mathbf{w}\|^2. \quad (9)$$

The β terms of equation (3) may be absorbed into \mathbf{A}_i by augmenting the features by vectors of ones, so no generality is lost in equation (9).

Let us pause to take note of the extra terms in the cost function (9). There is a term regularizing \mathbf{w} towards 0 and another regularizing \mathbf{r}_i towards \mathbf{q}_i . Vector \mathbf{q}_i is a “center” of regularization for the targets \mathbf{r}_i . If C_{r_i} are nonzero we set these to $\tilde{\mathbf{r}}_i$ when training scores are available, otherwise we use $\mathbf{q}_i = \text{Argmin}_{\mathbf{x}} \phi_i(\mathbf{x})$ when only ordering is available (this corresponds to $\mathbf{0}$ for square loss and uniform distribution for KL loss). This allows one to bias the targets towards the training scores when C_{r_i} is high and focus on order otherwise.

Proposition 1. *Let ϕ be s strongly convex with L Lipschitz continuous gradients, and σ_i be the smallest singular value of \mathbf{A}_i , then the cost function (9) is jointly convex if*

$$\sum \frac{\sigma_i(C_{w_i} + 1/L)}{|\mathcal{V}_i|} + \frac{C}{2} - \frac{4(\sum \frac{1}{|\mathcal{V}_i|})^2}{s \sum \frac{1+C_{r_i}}{|\mathcal{V}_i|}} \geq 0$$

Proof. $\sum_{i=1}^{|\mathcal{Q}|} \frac{1}{|\mathcal{V}_i|} \begin{bmatrix} (1+C_{r_i})H_\phi & -I \\ -I & \mathbf{A}_i^\dagger(H_\psi + C_{w_i})\mathbf{A}_i + \frac{C}{2}|\mathcal{Q}|I \end{bmatrix}$, is the Hessian of the cost function (9) where ψ is the Legendre conjugate of ϕ and H_ϕ, H_ψ the corresponding diagonal. Recall that $\phi(\cdot)$ and consequently $\psi(\cdot)$ are separable. The smallest eigenvalue of the Hessian may be bounded as the value of the following optimization problem:

$$\min \langle \mathbf{y}, \mathbf{y} \rangle \left(\sum_i \frac{\sigma_i}{|\mathcal{V}_i|} (C_{w_i} + \frac{1}{L}) + \frac{C}{2} \right) - 2 \langle \mathbf{x}, \mathbf{y} \rangle \sum_i \frac{|\mathcal{Q}|}{|\mathcal{V}_i|} + \langle \mathbf{x}, \mathbf{x} \rangle \sum_i \frac{s}{|\mathcal{V}_i|} (1 + C_{r_i}) \quad \text{s.t.} \quad \langle \mathbf{x}, \mathbf{x} \rangle + \langle \mathbf{y}, \mathbf{y} \rangle = 1 \quad (10)$$

where σ_i is the smallest singular value of \mathbf{A}_i . Invoking Cauchy-Schwarz inequality and treating the expression as a quadratic function in $\sqrt{\langle \mathbf{x}, \mathbf{x} \rangle}$ we can see that convexity is implied by $\sum \frac{\sigma_i(C_{w_i} + 1/L)}{|\mathcal{V}_i|} + \frac{C}{2} - \frac{4(\sum \frac{|\mathcal{Q}|}{|\mathcal{V}_i|})^2}{s \sum \frac{1+C_{r_i}}{|\mathcal{V}_i|}} \geq 0$ \square

Corollary 1. *The cost function (9) is jointly convex if $C \geq \frac{8\mathcal{Q}}{s(1+C_r)} (\sum \frac{1}{|\mathcal{V}_i|})$, if $C_{r_i} = C_r \forall_i$.*

Corollary 1 gives practitioners an easy thumb rule to ensure joint convexity.

These additional regularization terms do not come at an extra computational burden. Estimating \mathbf{r}, \mathbf{w} remain just as easy. We show that the result of the additional terms are that the \mathbf{r}_i updates (5) need to be computed with respect to the *deflected* predicted score $(\nabla \phi)^{-1}(\alpha \mathbf{A} \mathbf{w} + (1 - \alpha) \nabla \phi(\mathbf{q}_i))$, as opposed to the predicted score $(\nabla \phi)^{-1}(\mathbf{A} \mathbf{w})$.

Lemma 1. *Let $\alpha_i = \frac{1}{1+C_{r_i}}$, then $\text{Argmin}_{\mathbf{r}_i \in \mathcal{R}_{\downarrow_i} \cap \mathcal{S}_i} F_i(\mathbf{r}_i, \mathbf{w}) = \text{Argmin}_{\mathbf{r}_i \in \mathcal{R}_{\downarrow_i} \cap \mathcal{S}_i} D_\phi(\mathbf{r}_i) \left\| (\nabla \phi)^{-1}(\alpha_i \mathbf{A}_i \mathbf{w} + (1 - \alpha_i) \nabla \phi(\mathbf{q}_i)) \right\|^2$.*

Proof. Use $\mathbb{E}_{\mathbf{x} \sim \pi} [D_\phi(\mathbf{x} \parallel \mathbf{s})] = \mathbb{E}_{\mathbf{x} \sim \pi} [D_\phi(\mathbf{x} \parallel \boldsymbol{\mu})] + D_\phi(\boldsymbol{\mu} \parallel \mathbf{s})$ [4]. \square

3.1.2 Marginal Strong Convexity and Smoothness

Recall our motivations for pursuing joint convexity: (i) initialization independence of the training and (ii) more efficient training algorithms. In light of Proposition 1 and Corollary 1, the reader should be convinced of the former. In this section we explore how joint convexity may be exploited to provide an efficient optimization algorithm for training, as well as guarantees of convergence rates. Previous work on MR [1] come with no guarantees on rates of convergence.

The MR cost function was minimized in [1] using *exact* coordinate-wise minimizations. This can be expensive for the \mathbf{w}, β updates (7) because they are iterative in nature. Further since a single \mathbf{w}, β update is equivalent to solving a generalized linear model (GLM), it is clear that the MR procedure would be slower than solving for a GLM because typically multiple iterations of GLM update are required for convergence.

Here we will replace exact coordinate-wise minimizations over \mathbf{r}, \mathbf{w} by inexact gradient descent updates that satisfy any of the standard “sufficient descent” criteria [5] (for example Armijo’s criteria) used in gradient based methods. Joint convexity will play a crucial role in making this possible.

Joint convexity of $F(\{\mathbf{r}_i\}, \mathbf{w})$ allows us to work with the marginal function

$$G(\mathbf{w}) = \min_{\{\mathbf{r}_i\}} F(\{\mathbf{r}_i\}, \mathbf{w}) \quad (11)$$

without losing convexity. This luxury is not available in MR. The marginal function is guaranteed to be convex when the joint function is convex [23]. Recall convexity is always preserved under pointwise *maximization*, however if the function is *jointly convex* it is also preserved under pointwise minimization as in equation (11).

The gradient $\nabla G(\mathbf{w})$ of the marginal is obtained as

$$\nabla G(\mathbf{w}) = \sum_i^{|Q|} G_i(\mathbf{w}) = \sum_i^{|Q|} \nabla F_i(\{\mathbf{r}_i^*\}, \mathbf{w}) \quad (12)$$

where $\mathbf{r}_i^* = \text{Argmin}_{\mathbf{r}_i \in \mathcal{R}_i} F_i(\mathbf{r}_i, \mathbf{w})$.

Now we can make a few observations: for a choice of a closed form of $\phi(\cdot)$ we know ∇F_i in closed form. Hence the moment we are able to compute \mathbf{r}_i^* we can also compute the gradient of the function $G(\mathbf{w})$ and hence minimize it using any gradient based minimization methods. Also observe that this *gradient compu-*

tation trivially parallelizes because the \mathbf{r}_i s are all independent and can be computed simultaneously. We shall show that \mathbf{r}_i^* can be computed very efficiently in not only finite time but also linear in the number of training points per query. This is covered in Section 3.5.

If in addition to just convexity of the marginal function $G(\mathbf{w})$ we also had strong convexity, not only would it facilitate super-linear convergence of quasi-Newton methods, but it will also guarantee logarithmic regret in the online setting [11]. With these motivations in mind we investigate the conditions for strong convexity of $G(\mathbf{w})$. We do so by examining the Hessian $\nabla^2 G(\mathbf{w})$. Note however that $G(\mathbf{w})$ is not obtained in closed form but by equation (11), which we now need to differentiate twice to find the Hessian.

Differentiating Twice Under the Minimization

Sign: A prominent role is played in the analysis by the ability to differentiate under the minimization sign. We do not know the function $G(\mathbf{w})$ in closed form but are able to compute its Hessian in terms of \mathbf{r}_i^* . Using assumptions of continuous second order differentiability and the shorthand $F_i^* = F_i(\mathbf{r}_i^*, \mathbf{w})$ we obtain

$$\begin{aligned} \nabla^2 G_i(\mathbf{w}) &= \nabla_{\mathbf{w}}^2 F_i^* - \nabla_{\mathbf{w}, \mathbf{r}_i} \nabla F_i^{*\dagger} (\nabla_{\mathbf{r}_i}^2 F_i^*)^{-1} \nabla_{\mathbf{w}, \mathbf{r}_i} \nabla F_i^* = \\ &= \frac{\mathbf{A}_i^\dagger}{|\mathcal{V}_i|} [H_\psi + C_{w_i} - \frac{1}{1 + C_{r_i}} (H_\phi)^{-1}] \mathbf{A}_i + \frac{C}{|\mathcal{Q}|} I \end{aligned} \quad (13)$$

by differentiation twice under the min operator. Expression (13) will be useful because it allows to determine when is $G(\mathbf{w})$ strongly convex (see Lemma 2) and also because it gives us a way to compute the Hessian that is important for Newton methods that we employ.

Lemma 2. *If ϕ is s strongly convex with L -Lipschitz continuous gradient and σ_i is the principal singular value of A_i , then $G(\mathbf{w})$ is C strongly convex if $\sum_i \left(\frac{\sigma_i}{L} + \sigma_i C_{w_i} - \frac{\sigma_i}{s(1+C_{r_i})} \right) > 0$.*

Strong convexity and Lipschitz continuity of the gradient ensures that a gradient descent method will have linear rate of convergence [5]. Lemma 2 gives the practitioner a way to choose C_{w_i} and C_{r_i} appropriately.

Can the convergence rates be pushed further? Can we obtain locally quadratic convergence? We answer in the affirmative in the next section.

3.1.3 Lipschitz Continuity of Hessian

In order to enjoy local quadratic convergence, quasi-Newton methods require that the objective function (i) be twice differentiable, (ii) be strongly convex and (iii) have Lipschitz continuous Hessians [5]. The first two have already been established, now we explore

the third. Observe from equation (13) that we only need to be concerned about the sensitivity of the term $[H_\psi + C_{w_i} - \frac{1}{1+C_{\phi_i}}(H_\phi)^{-1}]$ to variations in \mathbf{w} . We make the notation more precise about dependency on \mathbf{w} . Let $\mathbf{r}_i^*(\mathbf{w}) = \text{Argmin}_{\mathbf{r}_i \in \mathcal{R}_i} F_i(\mathbf{r}_i, \mathbf{w})$ and the parenthesis indicate where the Hessians are evaluated in the expression: $[H_\psi(\mathbf{w}) + C_{w_i} - \frac{1}{1+C_{\phi_i}}(H_\phi(\mathbf{r}_i^*(\mathbf{w})))^{-1}]$.

Lemma 3. *Let $\psi(\cdot)$ be the Legendre conjugate of $\phi(\cdot)$ that defines the cost function $G(\mathbf{w})$ in equation (11). Then if $\psi(\cdot)$ has a Lipschitz continuous Hessian then $G(\mathbf{w})$ has a Lipschitz continuous Hessian.*

Proof. $[H_\psi(\mathbf{w}) + C_{w_i} - \frac{1}{1+C_{\phi_i}}(H_\phi(\mathbf{r}_i^*(\mathbf{w})))^{-1}] = [H_\psi(\mathbf{w}) + C_{w_i} - \frac{1}{1+C_{\phi_i}}H_\psi(\nabla\phi(\mathbf{r}_i^*(\mathbf{w})))]$ using Legendre duality. Further, the vector $\nabla\phi(\mathbf{r}_i^*(\mathbf{w}))$ turns out to be the Euclidean projection of the vector $\mathbf{A}_i\mathbf{w}$ on the set \mathcal{R}_i (see Proposition 2). Since projection is a non-expansive operator, $H_\psi(\nabla\phi(\mathbf{r}_i^*(\mathbf{w})))$ is Lipschitz continuous in \mathbf{w} . \square

3.1.4 Summary: Impact on Optimization

Let us take stock of what have we achieved so far. Lemmas 1 through 3 led to quantitative guarantees on rate of convergence in the batch setting. They allow selecting the regularization parameters C_{ϕ_i}, C_{w_i} based on desired convergence performance. The paper [1] could not provide any such quantitative guarantees, because their cost function was not proven to be jointly convex. Note that the nested minimization in the gradient computation trivially parallelizes. We shall see that each parallel task completes in finite time (Section 3.6). *Batch gradient descent on the marginal with (12) evaluated in parallel* converges linearly as a result of strong marginal convexity and smoothness [5]. *Stochastic gradient descent by sampling an index from (12)* also has linear rate of convergence (in an expected sense) [20]. *Quasi-Newton (and truncated Newton) methods with parallel evaluation of gradients* use the gradient computation (12) (and explicit Hessian (13) which has a simple diagonal structure) have superlinear convergence [5].

3.2 Online Algorithm for Learning Permutations

In this section we propose an online model for learning to rank where we have a varying set of items that need to be ordered in each round. The adversary, at round t provides the feature matrix \mathbf{A}_t of d_t items that it has ranked, but that order is not revealed till the learner responds with a “scoring vector” \mathbf{w}_t . The learner is then charged a cost of $G_t(\mathbf{w}_t)$ as defined in (11) according to any twice differentiable σ strongly

convex function ϕ_t with L Lipschitz continuous gradient. The order and the function ϕ_t is then revealed for the learner to use. The objective is to minimize the cumulative loss $\sum_t G_t(\mathbf{w}_t)$.

For the t^{th} gradient update we use the t^{th} term of the gradient (12) with a learning rate of $\frac{1}{\sigma t}$ as

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \frac{1}{\sigma t} F_t(\{\mathbf{r}_t^*\}, \mathbf{w})$$

where $\mathbf{r}_t^* = \text{Argmin}_{\mathbf{r}_t \in \mathcal{R}_t \cap \mathcal{S}_t} F_t(\mathbf{r}_t, \mathbf{w})$ and F_t is defined in (9).

Theorem 1. [11] *The online gradient algorithm applied in an online setting to a s strongly function that has L Lipschitz continuous gradients has regret $\mathcal{O}(\frac{L^2}{\sigma} \log T)$.*

Neither the algorithm nor the bound is new, what is novel though is that the ranking problem of such combinatorial nature can be transformed into a form, without loss in generality, that this algorithm can exploit.

Summary: This concludes what we have to say about the implications joint convexity of the cost function we propose. One can see that it leads to quantitative guarantees on rate of convergence in the batch setting and performance guarantees in the online and the adversarial setting. Now we turn our attention the next topic of this paper: large margins.

3.3 Margins

Performant classification loss functions such as hinge loss [22], logistic loss and exponential loss [9] continue to be active even after training error has fallen to zero. For MR such a margin like property is not only beneficial but also essential because otherwise the cost function is degenerate as may be verified by setting $\mathbf{w}, \beta = \mathbf{0}$. The necessity of this margin property is not mentioned in [1]. Here we take an explicit approach.

By controlling the margin we can also model the notion that errors at the top of the list are more severe than at the bottom. We achieve this by adding linear inequalities and terms. Therefore the properties of strong convexity and Lipschitz continuity of the gradient established in Section 3.1 continue to hold.

We incorporate the margin property in two alternative ways. We augment the cost function (9) by introducing a fixed margin (14) and alternatively a large margin variant (15). In addition to enforcing order in the target vector \mathbf{r}_i it enforces (for the fixed margin formulation) or encourages (for the large margin formulation) a gap between the target values of adjacently ordered items $r_{i,j}, r_{i,j+1}$. In the formulations (14), (15), the components of \mathbf{t}_i denote the gap between the adjacent

targets. In (14) the gaps are pre-specified. It is natural to specify a comparatively higher gap at the top. In (15) the gaps are not specified explicitly, but a reward c_i is awarded per unit gap.

The **fixed margin formulations** is posed in terms of positive pre-prescribed margins $t_{i,j}$ as follows:

$$\begin{aligned} \min_{\mathbf{r}_i, \mathbf{w}} \sum_{i=1}^{|\mathcal{Q}|} F_i(\mathbf{r}_i, \mathbf{w}) \\ r_{i,j+1} - r_{i,j} \geq t_{i,j} \quad \forall j \in [0, d_i - 1], \forall i \in [1, |\mathcal{Q}|] \\ r_{i,0} \geq t_{i,0} \quad \forall i \in [1, |\mathcal{Q}|] \end{aligned} \quad (14)$$

The **large margin formulations** are posed in terms of a vector of *rewards* \mathbf{c}_i associated with the vector of gaps $\mathbf{t}_i > \mathbf{0}$ as follows: for every query $q_i \in \mathcal{Q}$, solve:

$$\begin{aligned} \min_{\mathbf{r}_i, \mathbf{w}, \mathbf{t}_i} \sum_{i=1}^{|\mathcal{Q}|} F_i(\mathbf{r}_i, \mathbf{w}) - \langle \mathbf{c}_i, \mathbf{t}_i \rangle \\ r_{i,j+1} - r_{i,j} \geq t_{i,j} \geq 0 \quad \forall j \in [0, d_i - 1], \forall i \in [1, |\mathcal{Q}|] \\ r_{i,0} \geq t_{i,0} \quad \forall i \in [1, |\mathcal{Q}|], \end{aligned} \quad (15)$$

Note that the \mathbf{r}_i optimization is a Bregman projection problem. Furthermore, the \mathbf{r}_i 's are independent and therefore can be projected in parallel. Readers familiar with generalized linear models (GLM) will recognize that the optimization over \mathbf{w} is penalized maximum likelihood parameter estimation for GLMs. Since this procedure is standard, we focus on \mathbf{r} and \mathbf{t} only in the interest of space.

3.4 Bregman Projection on $\mathcal{R}_{\downarrow \mathbf{t}}$

Both the formulations (14) and (15) involve Bregman projections on $\mathcal{R}_{\downarrow \mathbf{t}}$. Elements of $\mathcal{R}_{\downarrow \mathbf{t}} \subset \mathbb{R}^n$ are not only sorted but also have separation between adjacent components, given by the vector \mathbf{t} . In this section we reduce it to a square Euclidean projection on $\text{Argmin}_{\mathbf{y} \in \mathcal{R}_{\downarrow \mathbf{t}}}$, hence removing the need to solve a non-linear optimization problem. It is quite remarkable that this is possible. For the reduction to hold we need additional assumptions of strong convexity and/or Lipschitz continuity. Consider the problem:

$$\min_{\mathbf{r}} D_{\phi}(\mathbf{r} \parallel (\nabla \phi)^{-1}(A\mathbf{w})) \text{ s.t. } \text{Adj-Diff}(\mathbf{r}) \leq \mathbf{t}. \quad (16)$$

If $\mathbf{t} = \mathbf{0}$ this is $\min_{\mathbf{r} \in \mathcal{R}_{\downarrow}} D_{\phi}(\mathbf{r} \parallel (\nabla \phi)^{-1}(A\mathbf{w}))$. When \mathbf{t} is component-wise strictly positive it imposes strict margin between adjacent components of \mathbf{r} .

Proposition 2. *Let $\phi(\cdot)$ be s strongly convex, then*

$$\begin{aligned} (\nabla \phi)^{-1}(\mathbf{z}^*) = \text{Argmin}_{\mathbf{r}} D_{\phi}(\mathbf{r} \parallel (\nabla \phi)^{-1}(A\mathbf{w})) + \langle \mathbf{v}, \mathbf{r} \rangle \\ \text{s.t. } \text{Adj-Diff}(\mathbf{r}) \leq \mathbf{t} \end{aligned} \quad (17)$$

where $\mathbf{z}^* = \text{Argmin}_{\mathbf{z}} \|\mathbf{z} - A\mathbf{w}\| + \langle \mathbf{v}, \mathbf{r} \rangle$ s.t. $\text{Adj-Diff}(\mathbf{z}) \leq \mathbf{t}$.

Proof. For the moment let us ignore the term $\langle \mathbf{v}, \mathbf{r} \rangle$. Let the set of points satisfying the KKT conditions for (16) be $\mathcal{A} = \left\{ \mathbf{r} \mid \begin{array}{l} \nabla \phi(\mathbf{r}) = A\mathbf{w} - \text{Adj-Diff}(\mathbf{r}) \\ \text{Adj-Diff}(\mathbf{r}) \leq \mathbf{t} \end{array} \right\}$, let us denote the KKT points of the optimization problem

$$\min_{\mathbf{z}} \|\mathbf{z} - A\mathbf{w}\| \text{ s.t. } \text{Adj-Diff}(\mathbf{z}) \leq \mathbf{t} \text{ by } \mathcal{B} =$$

$$\left\{ \mathbf{z} \mid \begin{array}{l} \mathbf{z} = A\mathbf{w} - \text{Adj-Diff}(\mathbf{z}) \\ \text{Adj-Diff}(\mathbf{z}) \leq \mathbf{t} \end{array} \right\} = \left\{ \begin{array}{l} \nabla \phi(\mathbf{r}) \\ \mathbf{r} \mid \begin{array}{l} \nabla \phi(\mathbf{r}) = A\mathbf{w} - \text{Adj-Diff}(\mathbf{r}) \\ \text{Adj-Diff}(\nabla \phi(\mathbf{r})) \leq \mathbf{t} \end{array} \end{array} \right\}.$$

From $r_{j+1} - r_j \geq t_j$ and strong convexity we have $\nabla \phi(r_{j+1}) - \nabla \phi(r_j) \geq st_j$ thus $\mathcal{A} \subset \mathcal{B}$. Complementary slackness conditions are also verified thus \mathcal{A}, \mathcal{B} are unique minimizers. The term $\langle \mathbf{v}, \mathbf{r} \rangle$ maintains the relation between \mathcal{A} and \mathcal{B} proving that the minima of the two problems coincide. \square

Proposition 3. *Let $\phi(\cdot)$ be strictly convex, $\mathbf{t} \leq \mathbf{0}$ and $\nabla \phi(\cdot)$ $\frac{1}{L}$ Lipschitz continuous, then minimizer \mathbf{z}^* of (17) is*

$$\mathbf{z}^* = \text{Argmin}_{\mathbf{z}} \|\mathbf{z} - A\mathbf{w}\| + \langle \mathbf{v}, \mathbf{r} \rangle \text{ s.t. } \text{Adj-Diff}(\mathbf{z}) \leq L\mathbf{t}.$$

Proof. Define \mathcal{A} and \mathcal{B} as before. From $\nabla \phi(r_{j+1}) - \nabla \phi(r_j) \geq Lt_j$ and Lipschitz continuity we have $r_{j+1} - r_j \geq t_j$ therefore $\mathcal{B} \subset \mathcal{A}$, but \mathcal{A} and \mathcal{B} are unique minimizers. Therefore the proposition holds. \square

The implications: of the propositions are, of course, that, for the optimization over \mathbf{r} , one only needs to implement the square loss variants of (14) and (15) because they are in correspondence with other Bregman divergences as long as the convex function is strongly convex or its gradient is Lipschitz continuous.

The final piece is to show that the reduced quadratic program (QP) is efficiently solvable. This is critical because it is required for the numerical evaluating the gradient (and Hessian) of $G(\mathbf{w})$ where we cannot afford the expense of a generic QP solver. We now show how the QP can be solved in linear time.

3.5 Pool Adjacent Violators Algorithm

The pool adjacent violators algorithm [10] solves

$$\min_{\mathbf{z}} \|\mathbf{z} - A\mathbf{w}\| \text{ s.t. } \text{Adj-Diff}_*(\mathbf{z}) \leq \mathbf{0} \quad (18)$$

called the isotonic regression problem. PAV is essentially a block coordinate ascent of the dual of (18). It runs in *finite time* and a straight-forward implementation scales as $\mathcal{O}(d^2)$ in the dimensions. Subsequently [10] observed that if implemented carefully it remarkably has complexity that is linear in d .

The nonlinear optimization problems (14) and (15) from (18). Fortunately, by a series of non-linear and linear change of variables one can reduce these problems to variations of the isotonic regression problem.

3.6 Decomposing the Margin Formulation

For a fixed \mathbf{w} , a plausible way to optimize (15) is to fix \mathbf{t}_i and optimize \mathbf{r}_i and alternate, keeping \mathbf{w} fixed. One may update \mathbf{w} once \mathbf{t}_i and \mathbf{r}_i converge. This clearly fails because the constraints couple \mathbf{r}_i and \mathbf{t}_i . However, we show that an affine transformation can not only correctly decompose the problem, but also that it separates out the problem out into versions of isotonic regression problems: namely isotonic regression with a lower-bound on the smallest r . Thus it adds another (scalar) constraint to the system $\text{Adj-Diff}(\mathbf{r}) \leq -\mathbf{t}$, where Adj-Diff is the adjacent-difference operator.

Because of the reduction properties shown in Propositions 2 and 3 to estimate \mathbf{r}_i in (15) one only needs to consider the problem of the form:

$$\min_{\mathbf{r}_i, \mathbf{t}_i} \frac{1}{2} \|\mathbf{r}_i - \mathbf{y}_i\|^2 - \langle \mathbf{c}_i, \mathbf{t}_i \rangle \text{ s.t. } \text{Adj-Diff}(\mathbf{r}_i) \leq -\mathbf{t}_i, \quad \mathbf{t}_i > 0.$$

Substituting $\mathbf{t}_i = -\text{Adj-Diff}(\mathbf{d}_i)$, $\mathbf{z}_i = \mathbf{r}_i - \mathbf{d}_i$ we obtain

$$\begin{aligned} & \frac{1}{2} \|\mathbf{z}_i + \mathbf{d}_i - \mathbf{y}_i\|^2 + \langle \mathbf{c}_i, \text{Adj-Diff}(\mathbf{d}_i) \rangle \\ & \text{s.t. } \text{Adj-Diff}(\mathbf{z}_i) \leq 0, \quad \text{Adj-Diff}(\mathbf{d}_i) \leq 0. \end{aligned} \quad (19)$$

The variables \mathbf{z}_i and \mathbf{d}_i are completely decoupled, the constraints are the ordering constraints, and if either \mathbf{z}_i or \mathbf{d}_i fixed, the formulation reduces to an isotonic problem in the other (for \mathbf{d}_i some simple algebraic manipulation is necessary to expose the PAV form). Thus, one may alternate over \mathbf{z}_i and \mathbf{d}_i as follows:

$$\mathbf{z}_i^{t+1} = \text{PAV}(\mathbf{y}_i - \mathbf{d}_i^t) \quad (20)$$

$$\mathbf{d}_i^{t+1} = \text{PAV}(\mathbf{y}_i - \mathbf{z}_i^{t+1} - \text{Adj-Diff}^\dagger(\mathbf{c})) \quad (21)$$

and obtain the large margin solution by recovering $\mathbf{r}_i, \mathbf{t}_i$ from converged \mathbf{z}_i and \mathbf{d}_i .

Problem (14) can be decomposed similarly using propositions 2, 3 and the exact same affine transformation $\mathbf{t}_i = -\text{Adj-Diff}(\mathbf{d}_i)$ and $\mathbf{z}_i = \mathbf{r}_i - \mathbf{d}_i$. Here however \mathbf{d}_i is immediately determined, so no iteration over the variables \mathbf{z}_i and \mathbf{d}_i is necessary and solving $\mathbf{z}_i = \text{PAV}(\mathbf{y}_i - \mathbf{d}_i)$ is sufficient to recover the optimal \mathbf{r}_i . *Since this requires a single instance of PAV, it is obvious that this converges in finite time, linear in the number of items.*

4 EXPERIMENTS

We evaluated the ranking performance of the proposed margin equipped monotone retargeting (MEMR) ap-

	Sqr.MEMR LBFGS	Sqr.MEMR TRON	Sqr MR	RankSVM
MQ'07	0.166s	0.101s	26.396s	17.187s
	KL.MEMR LBFGS	KL.MEMR TRON	KL MR	-
MQ'07	0.326s	0.199s	54.15s	

Table 1: CPU time of MEMR and Baselines

HyperThreads	1	2	3	4	8
Sqr.MEMR LBFGS _{ms}	166	91	72	59	46
Speedup	1	1.8	2.3	2.8	3.6

Table 2: MEMR speedup with parallelism

proach on the benchmark LETOR 4.0 datasets [18] as well as the OHSUMED dataset [12]. Each of these datasets are pre-partitioned into five-fold validation sets for easy comparison across algorithms. We focus on the variants that use Sqr-loss and KL-divergence because these are strongly convex Bregman divergences. We compare the performance of MEMR against the following strong baselines (i) The MR algorithm as reported in [1] (Recall that the MR algorithm has been shown to outperform many of the current state of the art techniques [1]), (ii) NDCG consistent generalized linear models that also use different Bregman divergences [21] and (iii) max-margin based pairwise learning to rank method RankSVM as implemented by SVMPerf [14] (Note RankSVM as implemented by SVMPerf is a factor of 20 faster than its original implementation in SVMLight). MEMR is implemented in C++ as a minimization method on the function $G(\mathbf{w})$. PAV algorithm is used to compute the gradient, and the Hessian. We tried two strategies (i) quasi-Newton using LBFGS [17] and (ii) Trust region truncated Newton (TRON) [16]. While both were an order of magnitude faster than our baselines the latter gave the fastest convergence. The CPU timings of serial implementations on a 2.8 Ghz Intel Quad core processor are reported in Table 1. We parallelized the LBFGS based implementation. The timings and corresponding speedups are shown in Table 2. We found that overprovisioning of threads (8 threads on a quad-core) was necessary to reach full speedup supported by the hardware.

In our experiments the fixed margin constraints (see equation (14)) were set using different non-increasing functions of the rank. In Figure 2 we show the effect of margins set to different constant values. In Figure 3 we show the effect of margins set by different polynomially decaying functions. The regularization parameter C was selected on the basis of maximum NDCG on

MQ 2007: Mean NDCG (non-truncated)			
	SQ	KL	Hinge
MEMR	0.7491	0.7564	-
MR	0.7398	0.6978	-
NDCG consistent GLM [21]	0.7344	0.7399	-
RankSVM	-	-	0.6528

Table 3: Test NDCG on MQ2007 Dataset

OHSUMED: Mean NDCG (non-truncated)			
	SQ	KL	Hinge
MEMR	0.7115	0.7146	-
MR	0.6878	0.6997	-
NDCG consistent GLM [21]	0.6892	0.6947	-
RankSVM	-	-	0.6571

Table 4: Test NDCG on OHSUMED Dataset.

the validation set. Figure 4 shows the behavior of the same margin function but for the loss measured by KL divergence.

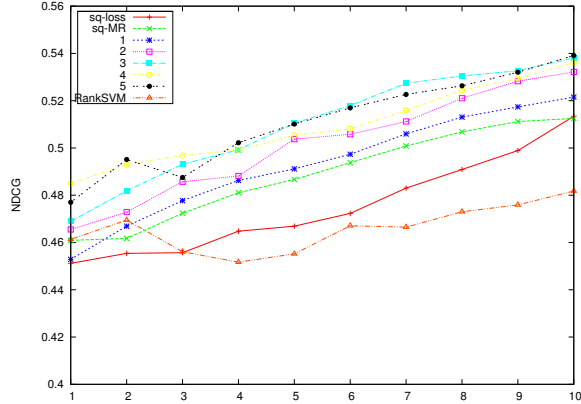


Figure 2: Truncated NDCG@N obtained on MQ2007 using Sqr-loss MEMR with margin between adjacent targets set to $\{0.0625e-3, 0.125e-3, 0.25e-3, .5e-3, 1e-3\}$ respectively showing improved rank quality as margins increase. The plot labeled “Sqr-Loss” represents pointwise NDCG consistent Sqr loss proposed by [21]. Plot labeled “Sqr-MR” corresponds to MR [1] with Sqr-loss. Performance of RankSVM is also shown

5 CONCLUSION

In this paper we presented a margin based monotone retargeting framework for learning to rank. Pointwise ranking methods search for optimal parameters of a regression function to fit the training scores that were specified to define the correct ranking order. MEMR on the other hand searches not only for optimal parameters of a regression function but also over all order-

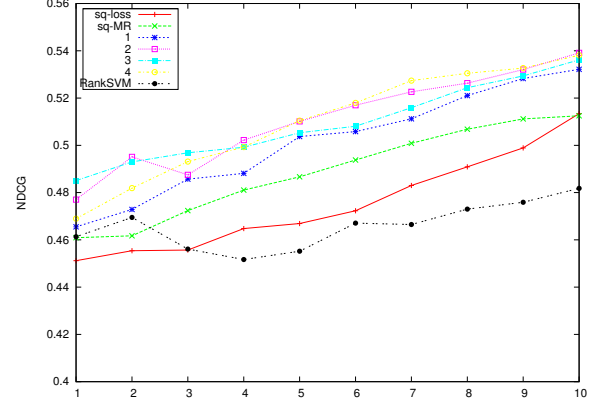


Figure 3: Truncated NDCG@N obtained on MQ2007 using Sqr-loss MEMR with margin between adjacent targets set by function $\frac{C}{\sqrt{r}}$ on the rank associated with the target. Plots shown for values of $C \in \{0.0625e-3, 0.125e-3, 0.25e-3, .5e-3\}$. The baselines are the same as in Figure 2.

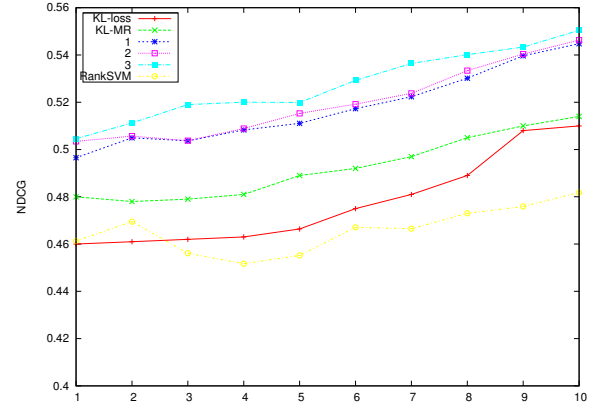


Figure 4: Truncated NDCG@N obtained on MQ2007 using KL-loss MEMR with margin between adjacent targets set by the function $\frac{C}{\sqrt{r}}$ for values of $C \in \{1e-1, 2e-1, 3e-1, 4e-1\}$. The plot labeled “KL-Loss” corresponds KL loss minimizing NDCG consistent GLM [21].

preserving transformations of the training score vectors such that its adjacent components are well separated. The separation property leads to state of the art performance as compared to MR and other max-margin based ranking formulations. Moreover its joint convexity and second order smoothness properties permit efficient algorithms that lead to running times that are a small fraction of competing algorithms, giving almost the best of both worlds: ranking accuracy better than pairwise methods and running times comparable to simple pointwise methods.

References

- [1] S. Acharyya, O. Koyejo, and J. Ghosh. Learning to rank with Bregman divergences and monotone retargeting. In *Uncertainty in Artificial Intelligence, UAI*, 2012.
- [2] Nir Ailon and Mehryar Mohri. An efficient reduction of ranking to classification. In *Conference on Learning Theory, COLT 2008*, pages 87–98, 2008.
- [3] Ricardo A. Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., 1999.
- [4] A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh. Clustering with Bregman divergences. *Journal of Machine Learning Research*, 6:1705–1749, 2005.
- [5] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
- [6] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: from pairwise approach to listwise approach. In *24th international conference on Machine learning, ICML’07*, 2007.
- [7] Olivier Chapelle, Donald Metzler, Ya Zhang, and Pierre Grinspan. Expected reciprocal rank for graded relevance. In *18th ACM conference on Information and knowledge management, CIKM ’09*, pages 621–630, 2009.
- [8] William. Cohen, Robert Schapire, and Yoram Singer. Learning to order things. *Journal of Artificial Intelligence Research*, 10:243–270, 1999.
- [9] Michael Collins, Robert Schapire, and Yoram Singer. Logistic regression, adaboost and Bregman distances. In Nicolo Cesa-Bianchi and Sally Goldman, editors, *COLT*, pages 158–169, 2000.
- [10] S.J. Grotzinger and C. Witzgall. Projections onto order simplexes. *Applied Mathematics and Optimization*, 12:247–270, 1984.
- [11] Elad Hazan, Amit Agarwal, and Satyen Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69:169–192, 2007.
- [12] William Hersh, Chris Buckley, T. J. Leone, and David Hickam. OHSUMED: an interactive retrieval evaluation and new large test collection for research. *SIGIR ’94*, pages 192–201, 1994.
- [13] Kalervo Järvelin and Jaana Kekäläinen. IR evaluation methods for retrieving highly relevant documents. In *23rd ACM SIGIR conference on Research and development in information retrieval, SIGIR ’00*, pages 41–48, 2000.
- [14] T. Joachims. Training linear SVMs in linear time. In *KDD’06 Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006.
- [15] Jyrki Kivinen and Manfred K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132, 1995.
- [16] Chih-Jen Lin, Ruby C. Weng, and S. Sathiya Keerthi. Trust region newton method for logistic regression. *J. Mach. Learn. Res.*, 9:627–650, June 2008.
- [17] Dong C. Liu, Jorge Nocedal, Dong C. Liu, and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45:503–528, 1989.
- [18] Tie-Yan Liu, Jun Xu, Tao Qin, Wenying Xiong, and Hang Li. LETOR: Benchmark dataset for research on learning to rank for information retrieval. In *SIGIR 2007 Workshop on Learning to Rank for Information Retrieval*, 2007.
- [19] C. E. McCulloch and S. R. Searle. *Generalized Linear and Mixed Models*. John Wiley & Sons, 2001.
- [20] Alexander Rakhlin, Ohad Shamir, and Karthik Sridharan. Making gradient descent optimal for strongly convex stochastic optimization. In *ICML*, 2012.
- [21] Pradeep Ravikumar, Ambuj Tewari, and Eunho Yang. On NDCG consistency of listwise ranking methods. In *Proceedings of 14th International Conference on Artificial Intelligence and Statistics, AISTATS*, 2011.
- [22] Mark Reid and Robert C Williamson. Information, divergence and risk for binary experiments. *Journal of Machine Learning Research*, 12:731–817, 2011.
- [23] R. T. Rockafellar. *Convex Analysis (Princeton Landmarks in Mathematics and Physics)*. Princeton University Press, December 1996.
- [24] Markus Weimer, Alexandros Karatzoglou, Quoc V. Le, and Alex J. Smola. CoFi Rank - maximum margin matrix factorization for collaborative ranking. In *NIPS*, 2007.
- [25] Jason Weston and John Blitzer. Latent structured ranking. In *Uncertainty in Artificial Intelligence, UAI 2012*, 2012.

On Convergence and Optimality of Best-Response Learning with Policy Types in Multiagent Systems

Stefano V. Albrecht
School of Informatics
University of Edinburgh
Edinburgh EH8 9AB, UK
s.v.albrecht@sms.ed.ac.uk

Subramanian Ramamoorthy
School of Informatics
University of Edinburgh
Edinburgh EH8 9AB, UK
s.ramamoorthy@ed.ac.uk

Abstract

While many multiagent algorithms are designed for homogeneous systems (i.e. all agents are identical), there are important applications which require an agent to coordinate its actions without knowing *a priori* how the other agents behave. One method to make this problem feasible is to assume that the other agents draw their latent policy (or type) from a specific set, and that a domain expert could provide a specification of this set, albeit only a partially correct one. Algorithms have been proposed by several researchers to compute posterior beliefs over such policy libraries, which can then be used to determine optimal actions. In this paper, we provide theoretical guidance on two central design parameters of this method: Firstly, it is important that the user choose a posterior which can learn the true distribution of latent types, as otherwise suboptimal actions may be chosen. We analyse convergence properties of two existing posterior formulations and propose a new posterior which can learn correlated distributions. Secondly, since the types are provided by an expert, they may be inaccurate in the sense that they do not predict the agents' observed actions. We provide a novel characterisation of optimality which allows experts to use efficient model checking algorithms to verify optimality of types.

1 INTRODUCTION

Many multiagent algorithms are developed with a homogeneous setting in mind, meaning that all agents use the same algorithm and are *a priori* aware of this fact. However, there are important applications for which this assumption may not be adequate, such as human-machine interaction, robot search and rescue, and financial markets. In such problems, it is important that an agent be able to effectively coordinate its actions without knowing *a priori* how the other agents

behave. The importance of this problem has been discussed in works such as [Albrecht and Ramamoorthy, 2013, Stone et al., 2010, Bowling and McCracken, 2005].

This problem is hard since the agents may exhibit a large variety of behaviours. General-purpose algorithms for multiagent learning are often impracticable, either because they take too long to produce effective policies or because they rely on prior coordination of behaviours [Albrecht and Ramamoorthy, 2012]. However, it has been recognised (e.g. [Albrecht and Ramamoorthy, 2013, Barrett et al., 2011]) that the complexity of this problem can often be reduced by assuming that there is a latent set of policies for each agent and a latent distribution over these policies, and that a domain expert can provide informed guesses as to what the policies might be. (These guesses could also be generated automatically, e.g. using some machine learning method on a corpus of historical data.)

One algorithm that takes this approach is *Harsanyi-Bellman Ad Hoc Coordination* (HBA) [Albrecht and Ramamoorthy, 2013]. This algorithm maintains a set of user-defined types (by “type”, we mean a policy or programme which specifies the behaviour of an agent) over which it computes posterior beliefs based on the agents' observed actions. The beliefs are then used in a planning procedure to compute expected payoffs for all actions (a procedure combining the concepts of Bayesian Nash equilibrium and Bellman optimality) and the best action is chosen. HBA was implemented as a reinforcement learning procedure and shown to be effective in both simulated and human-machine problems [Albrecht and Ramamoorthy, 2013]. Similar algorithms were studied in [Barrett et al., 2011, Carmel and Markovitch, 1999].

While works such as [Albrecht and Ramamoorthy, 2013, Barrett et al., 2011, Carmel and Markovitch, 1999] demonstrate the practical usefulness of such methods, they provide no theoretical guidance on two central design parameters: Firstly, one may compute the posterior beliefs in various ways, and it is important that the user choose a posterior formulation which is able to accurately approximate the latent distribution of types. This is important as otherwise the expected payoffs may be inaccurate, in which case HBA may

choose suboptimal actions. In this paper, we analyse the convergence conditions of two existing posterior formulations and we propose a new posterior which can learn correlated type distributions. These theoretical insights can be applied by the user to choose appropriate posteriors.

Secondly, since the types are provided by the user (or generated automatically), they may be inaccurate in the sense that their predictions deviate from the agents' observed actions. This raises the need for a theoretical analysis of how much and what kind of inaccuracy is acceptable for HBA to be able to solve its task, by which we mean that it drives the system into a terminal state. (A different question pertains to payoff maximisation; we focus on task accomplishment as it already includes many practical problems.) We describe a methodology in which we formulate a series of desirable termination guarantees and analyse the conditions under which they are met. Furthermore, we provide a novel characterisation of optimality which is based on the notion of probabilistic bisimulation [Larsen and Skou, 1991]. In addition to concisely defining what constitutes optimal type spaces, this allows the user to apply efficient model checking algorithms to verify optimality in practice.

2 RELATED WORK

Opponent modelling methods such as case-based reasoning [Gilboa and Schmeidler, 2001] and recursive modelling [Gmytrasiewicz and Durfee, 2000] are relevant to the extent that they can complement the user-defined types by creating new types (the opponent models) on the fly. For example, [Albrecht and Ramamoorthy, 2013] used a variant of case-based reasoning and [Barrett et al., 2011] used a tree-based classifier to complement the user-defined types.

Plays and play books [Bowling and McCracken, 2005] are similar in spirit to types and type spaces. However, plays specify the behaviour of an entire team, with additional structure such as applicability and termination conditions, and roles for each agent. In contrast, types specify the action probabilities of a single agent and do not require commitment to conditions and roles.

Plans and plan libraries [Carberry, 2001] are conceptually similar to types and type spaces. However, the focus of plan recognition has been on identifying the goal of an agent (e.g. [Bonchek-Dokow et al., 2009]) and efficient representation of plans (e.g. [Avrahami-Zilberbrand and Kaminka, 2007]), while types are used primarily to compute expected payoffs and can be efficiently represented as programmes [Albrecht and Ramamoorthy, 2013, Barrett et al., 2011].

I-POMDPs [Gmytrasiewicz and Doshi, 2005] and I-DIDs [Doshi et al., 2009] are related to our work since they too assume that agents have a latent type. These methods are designed to handle the full generality of partially observable states and latent types, and they explicitly model nested

beliefs. However, this generality comes at a high computational cost and the solutions are infeasible to compute in many cases. In contrast, we remain in the setting of fully observable states, and we implicitly allow for complex beliefs within the specification of types. This allows our methods to be computationally more tractable.

To the best of our knowledge, none of these related works directly address the theoretical questions considered in this paper. While our results apply to [Albrecht and Ramamoorthy, 2013, Barrett et al., 2011, Carmel and Markovitch, 1999], we believe they could be generalised to account for some of the other related works as well. This includes the methodology described in Section 5.

3 PRELIMINARIES

3.1 MODEL

Our analysis is based on the stochastic Bayesian game [Albrecht and Ramamoorthy, 2013]:

Definition 1. A *stochastic Bayesian game* (SBG) consists of

- discrete state space S with initial state $s^0 \in S$ and terminal states $\bar{S} \subset S$
- players $N = \{1, \dots, n\}$ and for each $i \in N$:
 - set of actions A_i (where $A = A_1 \times \dots \times A_n$)
 - type space Θ_i (where $\Theta = \Theta_1 \times \dots \times \Theta_n$)
 - payoff function $u_i : S \times A \times \Theta_i \rightarrow \mathbb{R}$
 - strategy $\pi_i : \mathbb{H} \times A_i \times \Theta_i \rightarrow [0, 1]$
- state transition function $T : S \times A \times S \rightarrow [0, 1]$
- type distribution $\Delta : \Theta \rightarrow [0, 1]$

where \mathbb{H} contains all *histories* $H^t = \langle s^0, a^0, s^1, a^1, \dots, s^t \rangle$ with $t \geq 0$, $(s^\tau, a^\tau) \in S \times A$ for $0 \leq \tau < t$, and $s^t \in S$.

Definition 2. A SBG starts at time $t = 0$ in state s^0 :

1. In state s^t , the types $\theta_1^t, \dots, \theta_n^t$ are sampled from Θ with probability $\Delta(\theta_1^t, \dots, \theta_n^t)$, and each player i is informed only about its own type θ_i^t .
2. Based on the history H^t , each player i chooses an action $a_i^t \in A_i$ with probability $\pi_i(H^t, a_i^t, \theta_i^t)$, resulting in the joint action $a^t = (a_1^t, \dots, a_n^t)$.
3. The game transitions into a successor state $s^{t+1} \in S$ with probability $T(s^t, a^t, s^{t+1})$, and each player i receives an individual payoff given by $u_i(s^t, a^t, \theta_i^t)$.

This process is repeated until a terminal state $s^t \in \bar{S}$ is reached, after which the game stops.

3.2 ASSUMPTIONS

We make the following general assumptions in our analysis:

Assumption 1. We control player i , by which we mean that we choose the strategies π_i (using HBA). Hence, player i has only one type, θ_i , which is known to us.

We sometimes omit θ_i in u_i and π_i for brevity, and we use j and $-i$ to refer to the other players (e.g. $A_{-i} = \times_{j \neq i} A_j$).

Assumption 2. Given a SBG Γ , we assume that all elements of Γ are known except for the type spaces Θ_j and the type distribution Δ , which are *latent variables*.

Assumption 3. We assume *full observability* of states and actions. That is, we are always informed of the current history H^t before making a decision.

Assumption 4. For any type θ_j and history H^t , there exists a *unique* sequence $(\chi_{a_j})_{a_j \in A_j}$ such that $\pi_j(H^t, a_j, \theta_j) = \chi_{a_j}$ for all $a_j \in A_j$.

We refer to this as *external* randomisation and to the opposite (when there is no unique χ_{a_j}) as *internal* randomisation. Technically, Assumption 4 is implied by the fact that π_j is a function, which means that any input is mapped to exactly one output. However, in practice this can be violated if randomisation is used “inside” a type implementation, hence it is worth stating it explicitly. Nonetheless, it can be shown that under full observability, external randomisation is equivalent to internal randomisation. Hence, Assumption 4 does not limit the types we can represent.

Example 1. Let there be two actions, A and B, and let the expected payoffs for agent i be $E(A) > E(B)$. The agent uses ϵ -greedy action selection [Sutton and Barto, 1998] with $\epsilon > 0$. If agent i randomises *externally*, then the strategy π_i will assign action probabilities $\langle 1 - \epsilon/2, \epsilon/2 \rangle$. If the agent randomises *internally*, then with probability ϵ it will assign probabilities $\langle 0.5, 0.5 \rangle$ and with probability $1 - \epsilon$ it will assign $\langle 1, 0 \rangle$, which is equivalent to external randomisation.

3.3 ALGORITHM

Algorithm 1 gives a formal definition of HBA (based on [Albrecht and Ramamoorthy, 2013]) which is the central algorithm in this analysis. (Section 1 provides an informal description.) Throughout this paper, we will use Θ_j^* and \Pr_j , respectively, to denote the user-defined type space and posterior for player j , where $\Pr_j(\theta_j^* | H^t)$ is the probability that player j has type $\theta_j^* \in \Theta_j^*$ after history H^t . Furthermore, we will use \Pr to denote the *combined* posterior, with $\Pr(\theta_{-i}^* | H^t) = \prod_{j \neq i} \Pr_j(\theta_j^* | H^t)$, and we sometimes refer to this simply as *the posterior*.

Note that the likelihood L in (1) is unspecified at this point. We will consider two variants for L in Section 4. The prior probabilities $P_j(\theta_j^*)$ in (1) can be used to specify prior beliefs about the distribution of types. It is convenient to specify $\Pr_j(\theta_j^* | H^t) = P_j(\theta_j^*)$ for $t = 0$. Finally, note that (2)/(3) define an infinite regress. In practice, this may be implemented using stochastic sampling (e.g. as in [Albrecht and Ramamoorthy, 2013, Barrett et al., 2011]) or by terminating the regress after some finite amount of time. In this analysis, we assume that (2)/(3) are implemented as given.

Algorithm 1 Harsanyi-Bellman Ad Hoc Coordination (HBA) [Albrecht and Ramamoorthy, 2013]

Input: SBG Γ , player i , user-defined type spaces Θ_j^* , history H^t , discount factor $0 \leq \gamma \leq 1$

Output: Action probabilities $\pi_i(H^t, a_i)$

1. For each $j \neq i$ and $\theta_j^* \in \Theta_j^*$, compute posterior probability

$$\Pr_j(\theta_j^* | H^t) = \frac{L(H^t | \theta_j^*) P_j(\theta_j^*)}{\sum_{\hat{\theta}_j^* \in \Theta_j^*} L(H^t | \hat{\theta}_j^*) P_j(\hat{\theta}_j^*)} \quad (1)$$

2. For each $a_i \in A_i$, compute expected payoff $E_{s^t}^{a_i}(H^t)$ with

$$E_s^{a_i}(\hat{H}) = \sum_{\theta_{-i}^* \in \Theta_{-i}^*} \Pr(\theta_{-i}^* | H^t) \sum_{a_{-i} \in A_{-i}} Q_s^{a_i, -i}(\hat{H}) \prod_{j \neq i} \pi_j(\hat{H}, a_j, \theta_j^*) \quad (2)$$

$$Q_s^a(\hat{H}) = \sum_{s' \in S} T(s, a, s') \left[u_i(s, a) + \gamma \max_{a_i} E_{s'}^{a_i}(\langle \hat{H}, a, s' \rangle) \right] \quad (3)$$

where $\Pr(\theta_{-i}^* | H^t) = \prod_{j \neq i} \Pr_j(\theta_j^* | H^t)$ and $a_{i, -i} \triangleq (a_i, a_{-i})$

3. Distribute $\pi_i(H^t, \cdot)$ uniformly over $\arg \max_{a_i} E_{s^t}^{a_i}(H^t)$

4 LEARNING THE TYPE DISTRIBUTION

This section is concerned with convergence and correctness properties of the posterior. The theorems in this section tell us if and under what conditions HBA will learn the type distribution of the game. As can be seen in Algorithm 1, this is important since the accuracy of the expected payoffs (2) depends crucially on the accuracy of the posterior (1).

However, for this to be a well-posed learning problem, we have to assume that the posterior \Pr can refer to the same elements as the type distribution Δ . Therefore, the results in this section pertain to a weaker form of *ad hoc coordination* [Albrecht and Ramamoorthy, 2013] in which the user knows that the latent type space Θ_j must be a subset of the user-defined type space Θ_j^* . Formally, we assume:

Assumption 5. $\forall j \neq i : \Theta_j \subseteq \Theta_j^*$

Based on this assumption, we simplify the notation in this section by dropping the $*$ in θ_j^* and Θ_j^* . The general case in which Assumption 5 does *not* hold is addressed in Section 5.

We consider two kinds of type distributions:

Definition 3. A type distribution Δ is called *pure* if there is $\theta \in \Theta$ such that $\Delta(\theta) = 1$. A type distribution is called *mixed* if it is not pure.

Pure type distributions can be used to model the fact that each player has a fixed type throughout the game, e.g. as in [Barrett et al., 2011]. Mixed type distributions, on the other hand, can be used to model randomly changing types. This

was shown in [Albrecht and Ramamoorthy, 2013], where a mixed type distribution was used to model defective agents and human behaviour.

4.1 PRODUCT POSTERIOR

We first consider the product posterior:

Definition 4. The *product posterior* is defined as (1) with

$$L(H^t|\theta_j) = \prod_{\tau=0}^{t-1} \pi_j(H^\tau, a_j^\tau, \theta_j) \quad (4)$$

This is the standard posterior formulation used in Bayesian games (e.g. [Kalai and Lehrer, 1993]) and was used in [Albrecht and Ramamoorthy, 2013, Barrett et al., 2011].

Our first theorem states that the product posterior is guaranteed to converge to any pure type distribution:

Theorem 1. Let Γ be a SBG with a pure type distribution Δ . If HBA uses a product posterior, and if the prior probabilities P_j are positive (i.e. $\forall \theta_j^* \in \Theta_j^* : P_j(\theta_j^*) > 0$), then, for $t \rightarrow \infty$: $\Pr(\theta_{-i}|H^t) = \Delta(\theta_{-i})$ for all $\theta_{-i} \in \Theta_{-i}$.

Proof. The proof is not difficult, but tedious. In the interest of space, we give a proof sketch.¹ [Kalai and Lehrer, 1993] studied a model which can be equivalently described as a single-state SBG ($|S| = 1$) with pure Δ and proved that the product posterior converges to the type distribution of the game. Their convergence result can be extended to multi-state SBGs by translating the multi-state SBG Γ into a single-state SBG $\tilde{\Gamma}$ which is equivalent to Γ in the sense that the players behave identically. Essentially, the trick is to remove the states in Γ by introducing a new player whose action choices correspond to the state transitions in Γ . \square

Theorem 1 states that the product posterior will learn any pure type distribution. However, it does not necessarily learn mixed type distributions, as shown in the following example:

Example 2. Consider a SBG with two players. Player 1 is controlled by HBA using a product posterior while player 2 has two types, θ_A and θ_B , which are assigned by a mixed type distribution Δ with $\Delta(\theta_A) = \Delta(\theta_B) = 0.5$. The type θ_A always chooses action A while θ_B always chooses action B. In this case, there will be a time t after which both types have been assigned at least once, and so both actions A and B have been played at least once by player 2. This means that from time t and all subsequent times $\tau \geq t$, we have $\Pr_2(\theta_A|H^\tau) = \Pr_2(\theta_B|H^\tau) = 0$ (since each type plays only one action), so the posterior will never converge to Δ .

4.2 SUM POSTERIOR

We now consider the sum posterior:

¹A full proof of Theorem 1 can be found at: <http://rad.inf.ed.ac.uk/data/publications/2014/uai14proof.pdf>

Definition 5. The *sum posterior* is defined as (1) with

$$L(H^t|\theta_j) = \sum_{\tau=0}^{t-1} \pi_j(H^\tau, a_j^\tau, \theta_j) \quad (5)$$

The sum posterior was introduced in [Albrecht and Ramamoorthy, 2013] to allow HBA to recognise changed types. In other words, the purpose of the sum posterior is to learn mixed type distributions. It is easy to see that a sum posterior would indeed learn the mixed type distribution in Example 2. However, we now give an example to show that without additional requirements the sum posterior does not necessarily learn any (pure or mixed) type distribution:

Example 3. Consider a SBG with two players. Player 1 is controlled by HBA using a sum posterior while player 2 has two types, θ_A and θ_{AB} , which are assigned by a pure type distribution Δ with $\Delta(\theta_A) = 1$. The type θ_A always chooses action A while θ_{AB} chooses actions A and B with equal probability. The product posterior converges to Δ , as predicted by Theorem 1. However, the sum posterior converges to probabilities $\langle \frac{2}{3}, \frac{1}{3} \rangle$, which is incorrect.

Note that this example can readily be modified to use a mixed type distribution, with similar results. Therefore, we conclude that the sum posterior does not necessarily learn any type distribution.

Under what condition is the sum posterior guaranteed to learn the true type distribution? Consider the following two quantities, which can be computed from a given history H^t :

Definition 6. The *average overlap* of player j in H^t is

$$AO_j(H^t) = \frac{1}{t} \sum_{\tau=0}^{t-1} [|\Lambda_j^\tau| \geq 2]_1 \sum_{\theta_j \in \Theta_j} \pi_j(H^\tau, a_j^\tau, \theta_j) |\Theta_j|^{-1} \quad (6)$$

$$\Lambda_j^\tau = \{\theta_j \in \Theta_j \mid \pi_j(H^\tau, a_j^\tau, \theta_j) > 0\}$$

where $[b]_1 = 1$ if b is true, else 0.

Definition 7. The *average stochasticity* of player j in H^t is

$$AS_j(H^t) = \frac{1}{t} \sum_{\tau=0}^{t-1} |\Theta_j|^{-1} \sum_{\theta_j \in \Theta_j} \frac{1 - \pi_j(H^\tau, \hat{a}_j^\tau, \theta_j)}{1 - |A_j|^{-1}} \quad (7)$$

where $\hat{a}_j^\tau \in \arg \max_{a_j} \pi_j(H^\tau, a_j, \theta_j)$.

Both quantities are bounded by 0 and 1. The average overlap describes the similarity of the types, where $AO_j(H^t) = 0$ means that player j 's types (on average) never chose the same action in history H^t , whereas $AO_j(H^t) = 1$ means that they behaved identically. The average stochasticity describes the uncertainty of the types, where $AS_j(H^t) = 0$ means that player j 's types (on average) were fully deterministic in the action choices in history H^t , whereas $AS_j(H^t) = 1$ means that they chose actions randomly with uniform probability.

We can show that, if the average overlap and stochasticity of player j converge to zero as $t \rightarrow \infty$, then the sum posterior is guaranteed to learn any pure or mixed type distribution:

Theorem 2. Let Γ be a SBG with a pure or mixed type distribution Δ . If HBA uses a sum posterior, then, for $t \rightarrow \infty$: If $AO_j(H^t) = 0$ and $AS_j(H^t) = 0$ for all players $j \neq i$, then $\Pr(\theta_{-i}|H^t) = \Delta(\theta_{-i})$ for all $\theta_{-i} \in \Theta_{-i}$.

Proof. Throughout this proof, let $t \rightarrow \infty$. The sum posterior is defined as (1) where L is defined as (5). Given the definition of L , both the numerator and the denominator in (1) may be infinite. We invoke L'Hôpital's rule which states that, in such cases, the quotient $\frac{u(t)}{v(t)}$ is equal to the quotient $\frac{u'(t)}{v'(t)}$ of the respective derivatives with respect to t . The derivative of L with respect to t is the average growth per time step, which in general may depend on the history H^t of states and actions. The average growth of L is

$$L'(H^t|\theta_j) = \sum_{a_j \in A_j} F(a_j|H^t) \pi_j(H^t, a_j, \theta_j) \quad (8)$$

where

$$F(a_j|H^t) = \sum_{\theta_j \in \Theta_j} \Delta(\theta_j) \pi_j(H^t, a_j, \theta_j) \quad (9)$$

is the probability of action a_j after history H^t , with $\Delta(\theta_j)$ being the marginal probability that player j is assigned type θ_j . As we will see shortly, we can make an asymptotic growth prediction irrespective of H^t . Given that $AO_j(H^t) = 0$, we can infer that whenever $\pi_j(H^t, a_j, \theta_j) > 0$ for action a_j and type θ_j , then $\pi_j(H^t, a_j, \theta'_j) = 0$ for all other types $\theta'_j \neq \theta_j$. Therefore, we can write (8) as

$$L'(H^t|\theta_j) = \Delta(\theta_j) \sum_{a_j \in A_j} \pi_j(H^t, a_j, \theta_j)^2 \quad (10)$$

Next, given that $AS_j(H^t) = 0$, we know that there exists an action a_j such that $\pi_j(H^t, a_j, \theta_j) = 1$, and therefore we can conclude that $L'(H^t|\theta_j) = \Delta(\theta_j)$. This shows that the history H^t is irrelevant to the asymptotic growth rate of L . Finally, since $\sum_{\theta_j \in \Theta_j} \Delta(\theta_j) = 1$, we know that the denominator in (1) will be 1, and we can ultimately conclude that $\Pr_j(\theta_j|H^t) = \Delta(\theta_j)$. \square

Theorem 2 explains why the sum posterior converges to the correct type distribution in Example 2. Since the types θ_A and θ_B always choose different actions and are completely deterministic (i.e. the average overlap and stochasticity are always zero), the sum posterior is guaranteed to converge to the type distribution. On the other hand, in Example 3 the types θ_A and θ_{AB} produce an overlap whenever action A is chosen, and θ_{AB} is completely random. Therefore, the average overlap and stochasticity are always positive, and an incorrect type distribution was learned.

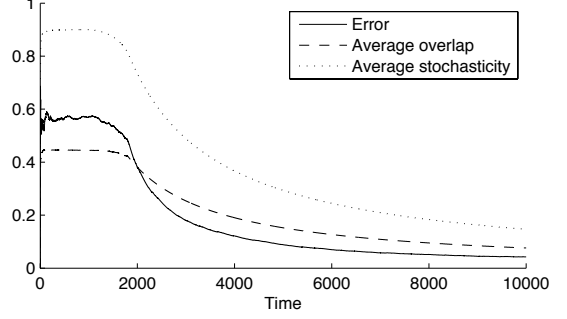


Figure 1: Example run in random SBG with 2 players, 10 actions, and 100 states. Player j has 3 reinforcement learning types with ϵ -greedy action selection (decreasing linearly from $\epsilon = 0.7$ at $t = 1000$, to $\epsilon = 0$ at $t = 2000$). The error at time t is $\sum_{\theta_j} |\Pr_j(\theta_j|H^t) - \Delta(\theta_j)|$ where \Pr_j is the sum posterior.

The assumptions made in Theorem 2, namely that the average overlap and stochasticity converge to zero, require practical justification. First of all, it is important to note that it is only required that these converge to zero *on average* as $t \rightarrow \infty$. This means that in the beginning there may be arbitrary overlap and stochasticity, as long as these go to zero as the game proceeds. In fact, with respect to stochasticity, this is precisely how the exploration-exploitation dilemma [Sutton and Barto, 1998] is solved in practice: In the early stages, the agent randomises deliberately over its actions in order to obtain more information about the environment (*exploration*) while, as the game proceeds, the agent becomes gradually more deterministic in its action choices so as to maximise its payoffs (*exploitation*). Typical mechanisms which implement this are ϵ -greedy and Softmax/Boltzmann exploration [Sutton and Barto, 1998]. Figure 1 demonstrates this in a SBG in which player j has 3 reinforcement learning types. The payoffs for the types were such that the average overlap would eventually go to zero.

Regarding the average overlap converging to zero, we believe that this is a property which should be guaranteed by *design*, for the following reason: If the user-defined type space Θ_j^* is such that there is a constantly high average overlap, then this means that the types $\theta_j^* \in \Theta_j^*$ are in effect very similar. However, types which are very similar are likely to produce very similar trajectories in the planning step of HBA (cf. \hat{H} in (2)) and, therefore, constitute redundancy in both time and space. Therefore, we believe it is advisable to use type spaces which have low average overlap.

4.3 CORRELATED POSTERIOR

An implicit assumption in the definition of (1) is that the type distribution Δ can be represented as a product of n independent factors (one for each player), so that $\Delta(\theta) = \prod_j \Delta_j(\theta_j)$. Therefore, since the sum posterior is in the form of (1), it is in fact only guaranteed to learn *independent* type distributions. This is opposed to *correlated* type distributions, which cannot be represented as a product

of n independent factors. Correlated type distributions can be used to specify constraints on type combinations, such as “player j can only have type θ_j if player k has type θ_k ”. The following example demonstrates how the sum posterior fails to converge to a correlated type distribution:

Example 4. Consider a SBG with 3 players. Player 1 is controlled by HBA using a sum posterior. Players 2 and 3 each have two types, θ_A and θ_B , which are defined as in Example 2. The type distribution Δ chooses types with probabilities $\Delta(\theta_A, \theta_B) = \Delta(\theta_B, \theta_A) = 0.5$ and $\Delta(\theta_A, \theta_A) = \Delta(\theta_B, \theta_B) = 0$. In other words, player 2 can never have the same type as player 3. From the perspective of HBA, each type (and hence action) is chosen with equal probability for both players. Thus, despite the fact that there is zero overlap and stochasticity, the sum posterior will eventually assign probability 0.25 to all constellations of types, which is incorrect. This means that HBA fails to recognise that the other players never choose the same action.

In this section, we propose a new posterior which can learn any correlated type distribution:

Definition 8. The *correlated posterior* is defined as

$$\Pr(\theta_{-i}|H^t) = \eta P(\theta_{-i}) \sum_{\tau=0}^{t-1} \prod_{\theta_j \in \theta_{-i}} \pi_j(H^\tau, a_j^\tau, \theta_j) \quad (11)$$

where P specifies prior probabilities (or beliefs) over Θ_{-i} (analogous to P_j) and η is a normalisation constant.

The correlated posterior is closely related to the sum posterior. In fact, it converges to the true type distribution under the same conditions as the sum posterior:

Theorem 3. Let Γ be a SBG with a *correlated* type distribution Δ . If HBA uses the correlated posterior, then, for $t \rightarrow \infty$: If $\text{AO}_j(H^t) = 0$ and $\text{AS}_j(H^t) = 0$ for all players $j \neq i$, then $\Pr(\theta_{-i}|H^t) = \Delta(\theta_{-i})$ for all $\theta_{-i} \in \Theta_{-i}$.

Proof. Proof is analogous to proof of Theorem 2. \square

It is easy to see that the correlated posterior would learn the correct type distribution in Example 4. Note that, since it is guaranteed to learn any correlated type distribution, it is also guaranteed to learn any independent type distribution. Therefore, the correlated posterior would also learn the correct type distribution in Example 2. This means that the correlated posterior is *complete* in the sense that it covers the entire spectrum of pure/mixed and independent/correlated type distributions. However, this completeness comes at a higher computational complexity. While the sum posterior is in $O(n \max_j |\Theta_j|)$ time and space, the correlated posterior is in $O(\max_j |\Theta_j|^n)$ time and space. In practice, however, the time complexity can be reduced drastically by computing the probabilities $\pi_j(H^\tau, a_j^\tau, \theta_j)$ only once for each j and $\theta_j \in \Theta_j$ (as in the sum posterior), and then reusing them in subsequent computations.

5 INACCURATE TYPE SPACES

Each user-defined type θ_j^* in Θ_j^* is a hypothesis by the user regarding how player j might behave. Therefore, Θ_j^* may be *inaccurate* in the sense that none of the types therein accurately predict the observed behaviour of player j . This is demonstrated in the following example:

Example 5. Consider a SBG with two players and actions L and R. Player 1 is controlled by HBA while player 2 has a single type, θ_{LR} , which chooses L,R,L,R, etc. HBA is provided with $\Theta_j^* = \{\theta_R^*, \theta_{LRR}^*\}$, where θ_R^* always chooses R while θ_{LRR}^* chooses L,R,L,R,R etc. Both user-defined types are inaccurate in the sense that they predict player 2’s actions in only $\approx 50\%$ of the game.

Two important theoretical questions in this context are how closely the user-defined type spaces Θ_j^* have to approximate the real type spaces Θ_j in order for HBA to be able to (1) solve the task (i.e. bring the SBG into a terminal state), and (2) achieve maximum payoffs. These questions are closely related to the notions of *flexibility* and *efficiency* [Albrecht and Ramamoorthy, 2013] which, respectively, correspond to the probability of termination and the average payoff per time step. In this section, we are primarily concerned with question 1, and we are concerned with question 2 only in so far as that we want to solve the task in minimal time. (Since reducing the time until termination will increase the average payoff per time step, i.e. increase efficiency.) This focus is formally captured by the following assumption, which we make throughout this section:

Assumption 6. Let player i be controlled by HBA, then $u_i(s, a, \theta_i) = 1$ iff. $s \in \tilde{S}$, else 0.

Assumption 6 specifies that we are only interested in reaching a terminal state, since this is the only way to obtain a non-zero payoff. In our analysis, we consider discount factors γ (cf. Algorithm 1) with $\gamma = 1$ and $\gamma < 1$. While all our results hold for both cases, there is an important distinction: If $\gamma = 1$, then the expected payoffs (2) correspond to the actual probability that the following state can lead to (or is) a terminal state (we call this the *success rate*), whereas this is not necessarily the case if $\gamma < 1$. This is since $\gamma < 1$ tends to prefer shorter paths, which means that actions with lower success rates may be preferred if they lead to faster termination. Therefore, if $\gamma = 1$ then HBA is solely interested in termination, and if $\gamma < 1$ then it is interested in *fast* termination, where lower γ prefers faster termination.

5.1 METHODOLOGY OF ANALYSIS

Given a SBG Γ , we define the *ideal process*, X , as the process induced by Γ in which player i is controlled by HBA and in which HBA always knows the current and all future types of all players. Then, given a posterior \Pr and user-defined type spaces Θ_j^* for all $j \neq i$, we define the *user process*, Y , as the process induced by Γ in which player i

is controlled by HBA (same as in X) and in which HBA uses Pr and Θ_j^* in the usual way. Thus, the only difference between X and Y is that X can always predict the player types whereas Y approximates this knowledge through Pr and Θ_j^* . We write $E_{s^t}^{a_i}(H^t|C)$ to denote the expected payoff (as defined by (2)) of action a_i in state s^t after history H^t , in process $C \in \{X, Y\}$.

The idea is that X constitutes the ideal solution in the sense that $E_{s^t}^{a_i}(H^t|X)$ corresponds to the *actual* expected payoff, which means that HBA chooses the truly best-possible actions in X . This is opposed to $E_{s^t}^{a_i}(H^t|Y)$, which is merely the *estimated* expected payoff based on Pr and Θ_j^* , so that HBA may choose suboptimal actions in Y . The methodology of our analysis is to specify what relation Y must have to X to satisfy certain guarantees for termination.

We specify such guarantees in PCTL [Hansson and Jonsson, 1994], a probabilistic modal logic which also allows for the specification of time constraints. PCTL expressions are interpreted over infinite histories in labelled transition systems with atomic propositions (i.e. Kripke structures). In order to interpret PCTL expressions over X and Y , we make the following modifications without loss of generality: Firstly, any terminal state $\bar{s} \in \bar{S}$ is an *absorbing* state, meaning that if a process is in \bar{s} , then the next state will be \bar{s} with probability 1 and all players receive a zero payoff. Secondly, we introduce the atomic proposition `term` and label each terminal state with it, so that `term` is true in s if and only if $s \in \bar{S}$.

We will use the following two PCTL expressions:

$$F_{>p}^{\leq t} \text{term}, F_{>p}^{< \infty} \text{term}$$

where $t \in \mathbb{N}$, $p \in [0, 1]$, and $> \in \{>, \geq\}$.

$F_{>p}^{\leq t} \text{term}$ specifies that, given a state s , with a probability of $> p$ a state s' will be reached from s within t time steps such that s' satisfies `term`. The semantics of $F_{>p}^{< \infty} \text{term}$ are similar except that s' will be reached in arbitrary but finite time. We write $s \models_C \phi$ to say that a state s satisfies the PCTL expression ϕ in process $C \in \{X, Y\}$.

5.2 CRITICAL TYPE SPACES

In the following section, we sometimes assume that the user-defined type spaces Θ_j^* are *uncritical*:

Definition 9. The user-defined type spaces Θ_j^* are *critical* if there is $S^c \subseteq S \setminus \bar{S}$ which satisfies:

1. For each $H^t \in \mathbb{H}$ with $s^t \in S^c$, there is $a_i \in A_i$ such that $E_{s^t}^{a_i}(H^t|Y) > 0$ and $E_{s^t}^{a_i}(H^t|X) > 0$
2. There is a positive probability that Y may eventually get into a state $s^c \in S^c$ from the initial state s^0
3. If Y is in a state in S^c , then with probability 1 it will always be in a state in S^c (i.e. it will never leave S^c)

We say Θ_j^* are *uncritical* if they are not critical.

Intuitively, critical type spaces have the potential to lead HBA into a state space in which it *believes* it chooses the right actions to solve the task, while other actions are *actually* required to solve the task. The only effect that its actions have is to induce an infinite cycle, due to a critical inconsistency between the user-defined and true type spaces. The following example demonstrates this:

Example 6. Recall Example 5 and let the task be to choose the same action as player j . Then, Θ_j^* is uncritical because HBA will always solve the task at $t = 1$, regardless of the posterior and despite the fact that Θ_j^* is inaccurate. Now, assume that $\Theta_j^* = \{\theta_{RL}^*\}$ where θ_{RL}^* chooses actions R,L,R,L etc. Then, Θ_j^* is critical since HBA will always choose the opposite action of player j , thinking that it would solve the task, when a different action would actually solve it.

A practical way to ensure that the type spaces Θ_j^* are (eventually) uncritical is to include methods for opponent modelling in each Θ_j^* (e.g. as in [Albrecht and Ramamoorthy, 2013, Barrett et al., 2011]). If the opponent models are guaranteed to learn the correct behaviours, then the type spaces Θ_j^* are guaranteed to become uncritical. In Example 6, any standard modelling method would eventually learn that the true strategy of player j is θ_{LR} . As the model becomes more accurate, the posterior gradually shifts towards it and eventually allows HBA to take the right action.

5.3 TERMINATION GUARANTEES

Our first guarantee states that if X has a positive probability of solving the task, then so does Y :

Property 1. $s^0 \models_X F_{>0}^{< \infty} \text{term} \Rightarrow s^0 \models_Y F_{>0}^{< \infty} \text{term}$

We can show that Property 1 holds if the user-defined type spaces Θ_j^* are uncritical and if Y only chooses actions for player i with positive expected payoff in X .

Let $\mathbb{A}(H^t|C)$ denote the set of actions that process C may choose from in state s^t after history H^t , i.e. $\mathbb{A}(H^t|C) = \arg \max_{a_i} E_{s^t}^{a_i}(H^t|C)$ (cf. step 3 in Algorithm 1).

Theorem 4. Property 1 holds if Θ_j^* are uncritical and

$$\forall H^t \in \mathbb{H} \forall a_i \in \mathbb{A}(H^t|Y) : E_{s^t}^{a_i}(H^t|X) > 0 \quad (12)$$

Proof. Assume $s^0 \models_X F_{>0}^{< \infty} \text{term}$. Then, we know that X chooses actions a_i which may lead into a state s' such that $s' \models_X F_{>0}^{< \infty} \text{term}$, and the same holds for all such states s' . Now, given (12) it is tempting to infer the same result for Y , since Y only chooses actions a_i which have positive expected payoff in X and, therefore, could truly lead into a terminal state. However, (12) alone is not sufficient to infer $s' \models_Y F_{>0}^{< \infty} \text{term}$ because of the special case in which Y chooses actions a_i such that $E_{s^t}^{a_i}(H^t|X) > 0$ but without ever reaching a terminal state. This is why we require that the user-defined type spaces Θ_j^* are uncritical, which prevents this special case. Thus, we can infer that $s' \models_Y F_{>0}^{< \infty} \text{term}$, and hence Property 1 holds. \square

The second guarantee states that if X always solves the task, then so does Y :

Property 2. $s^0 \models_X F_{\geq 1}^{\leq \infty} \text{term} \Rightarrow s^0 \models_Y F_{\geq 1}^{\leq \infty} \text{term}$

We can show that Property 2 holds if the user-defined type spaces Θ_j^* are uncritical and if Y only chooses actions for player i which lead to states into which X may get as well.

Let $\mu(H^t, s|C)$ be the probability that process C transitions into state s from state s^t after history H^t , i.e. $\mu(H^t, s|C) = \frac{1}{|\mathbb{A}|} \sum_{a_i \in \mathbb{A}} \sum_{a_{-i}} T(s^t, \langle a_i, a_{-i} \rangle, s) \prod_j \pi_j(H^t, a_j, \theta_j^t)$ with $\mathbb{A} \equiv \mathbb{A}(H^t|C)$, and let $\mu(H^t, S'|C) = \sum_{s \in S'} \mu(H^t, s|C)$ for $S' \subseteq S$.

Theorem 5. Property 2 holds if Θ_j^* are uncritical and

$$\forall H^t \in \mathbb{H} \forall s \in S : \mu(H^t, s|Y) > 0 \Rightarrow \mu(H^t, s|X) > 0 \quad (13)$$

Proof. The fact that $s^0 \models_X F_{\geq 1}^{\leq \infty} \text{term}$ means that, throughout the process, X only transitions into states s with $s \models_X F_{\geq 1}^{\leq \infty} \text{term}$. As before, it is tempting to infer the same result for Y based on (13), since it only transitions into states which have maximum success rate in X . However, (13) alone is not sufficient since Y may choose actions such that (13) holds true but Y will never reach a terminal state. Nevertheless, since the user-defined type spaces Θ_j^* are uncritical, we know that this special case will not occur, and hence Property 2 holds. \square

We note that, in both Properties 1 and 2, the reverse direction holds true regardless of Theorems 4 and 5. Furthermore, we can combine the requirements of Theorems 4 and 5 to ensure that both properties hold.

The next guarantee subsumes the previous guarantees by stating that X and Y have the same minimum probability of solving the task:

Property 3. $s^0 \models_X F_{\geq p}^{\leq \infty} \text{term} \Rightarrow s^0 \models_Y F_{\geq p}^{\leq \infty} \text{term}$

We can show that Property 3 holds if the user-defined type spaces Θ_j^* are uncritical and if Y only chooses actions for player i which X might have chosen as well.

Let $R(a_i, H^t|C)$ be the *success rate* of action a_i , formally $R(a_i, H^t|C) = E_{s^t}^{a_i}(H^t|C)$ with $\gamma = 1$ (so that it corresponds to the actual *probability* with which a_i may lead to termination in the future). Define X_{\min} and X_{\max} to be the processes which for each H^t choose actions $a_i \in \mathbb{A}(H^t|X)$ with, respectively, minimal and maximal success rate $R(a_i, H^t|X)$.

Theorem 6. If Θ_j^* are uncritical and

$$\forall H^t \in \mathbb{H} : \mathbb{A}(H^t|Y) \subseteq \mathbb{A}(H^t|X) \quad (14)$$

then

(i) for $\gamma = 1$: Proposition 3 holds in both directions

(ii) for $\gamma < 1$: $s^0 \models_X F_{\geq p}^{\leq \infty} \text{term} \Rightarrow s^0 \models_Y F_{\geq p'}^{\leq \infty} \text{term}$

with $p_{\min} \leq q \leq p_{\max}$ for $q \in \{p, p'\}$, where p_{\min} and p_{\max} are the highest probabilities such that $s^0 \models_{X_{\min}} F_{\geq p_{\min}}^{\leq \infty} \text{term}$ and $s^0 \models_{X_{\max}} F_{\geq p_{\max}}^{\leq \infty} \text{term}$.

Proof. (i): Since $\gamma = 1$, all actions $a_i \in \mathbb{A}(H^t|X)$ have the same success rate for a given H^t , and given (14) we know that Y 's actions always have the same success rate as X 's actions. Provided that the type spaces Θ_j^* are uncritical, we can conclude that Property 3 must hold, and for the same reasons the reverse direction must hold as well.

(ii): Since $\gamma < 1$, the actions $a_i \in \mathbb{A}(H^t|X)$ may have different success rates. The lowest and highest chances that X solves the task are precisely modelled by X_{\min} and X_{\max} , and given (14) and the fact that Θ_j^* are uncritical, the same holds for Y . Therefore, we can infer the common bound $p_{\min} \leq \{p, p'\} \leq p_{\max}$ as defined in Theorem 6. \square

Properties 1 to 3 are *indefinite* in the sense that they make no restrictions on time requirements. Our fourth and final guarantee subsumes all previous guarantees and states that if there is a probability p such that X terminates *within* t time steps, then so does Y for the same p and t :

Property 4. $s^0 \models_X F_{\geq p}^{\leq t} \text{term} \Rightarrow s^0 \models_Y F_{\geq p}^{\leq t} \text{term}$

We believe that Property 4 is an adequate criterion of optimality for the type spaces Θ_j^* since, if it holds, Θ_j^* must approximate Θ_j in a way which allows HBA to plan (almost) as accurately — in terms of solving the task — as the “ideal” HBA in X which always knows the true types.

What relation must Y have to X to satisfy Property 4? The fact that Y and X are processes over state transition systems means we can draw on methods from the model checking literature to answer this question. Specifically, we will use the concept of *probabilistic bisimulation* [Larsen and Skou, 1991], which we here define in the context of our work:

Definition 10. A *probabilistic bisimulation* between X and Y is an equivalence relation $B \subseteq S \times S$ such that

- (i) $(s^0, s^0) \in B$
- (ii) $s_X \models_X \text{term} \Leftrightarrow s_Y \models_Y \text{term}$ for all $(s_X, s_Y) \in B$
- (iii) $\mu(H_X^t, \hat{S}|X) = \mu(H_Y^t, \hat{S}|Y)$ for any histories H_X^t, H_Y^t with $(s_X^t, s_Y^t) \in B$ and all equivalence classes \hat{S} under B .

Intuitively, a probabilistic bisimulation states that X and Y do (on average) match each other's transitions. Our definition of probabilistic bisimulation is most general in that it does not require that transitions are matched by the same action or that related states satisfy the same atomic propositions other than termination. However, we do note that other definitions exist that make such additional requirements, and our results hold for each of these refinements.

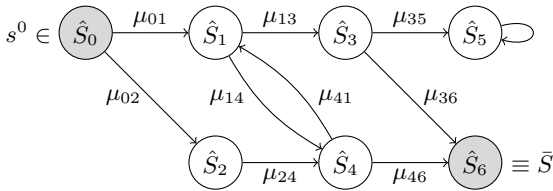
The main contribution of this section is to show that the

optimality criterion expressed by Property 4 holds in *both directions* if there is a probabilistic bisimulation between X and Y . Thus, we offer an alternative formal characterisation of optimality for the user-defined type spaces Θ_j^* :

Theorem 7. Property 4 holds in both directions if there is a probabilistic bisimulation between X and Y .

Proof. First of all, we note that, strictly speaking, the standard definitions of bisimulation (e.g. [Baier, 1996, Larsen and Skou, 1991]) assume the Markov property, which means that the next state of a process depends only on the current state of the process. In contrast, we consider the more general case in which the next state may depend on the history H^t of previous states and joint actions (since the player strategies π_j depend on H^t). However, one can always enforce the Markov property *by design*, i.e. by augmenting the state space S to account for the relevant factors of the past. In fact, we could postulate that the histories as a whole constitute the states of the system, i.e. $S = \mathbb{H}$. Therefore, to simplify the exposition, we assume the Markov property and we write $\mu(s, \hat{S}|C)$ to denote the cumulative probability that C transitions from state s into any state in \hat{S} .

Given the Markov property, the fact that B is an equivalence relation, and $\mu(s_X, \hat{S}|X) = \mu(s_Y, \hat{S}|Y)$ for $(s_X, s_Y) \in B$, we can represent the dynamics of X and Y in a common graph, such as the following one:



The nodes correspond to the equivalence classes under B . A directed edge from \hat{S}_a to \hat{S}_b specifies that there is a positive probability $\mu_{ab} = \mu(s_X, \hat{S}_b|X) = \mu(s_Y, \hat{S}_b|Y)$ that X and Y transition from states $s_X, s_Y \in \hat{S}_a$ to states $s'_X, s'_Y \in \hat{S}_b$. Note that s_X, s_Y and s'_X, s'_Y need not be equal but merely equivalent, i.e. $(s_X, s_Y) \in B$ and $(s'_X, s'_Y) \in B$. There is one node (\hat{S}_0) that contains the initial state s^0 and one node (\hat{S}_6) that contains all terminal states \bar{S} and no other states. This is because once X and Y reach a terminal state they will always stay in it (i.e. $\mu(s, \bar{S}|X) = \mu(s, \bar{S}|Y) = 1$ for $s \in \bar{S}$) and since they are the only states that satisfy `term`. Thus, the graph starts in \hat{S}_0 and terminates (if at all) in \hat{S}_6 .

Since the graph represents the dynamics of both X and Y , it is easy to see that Property 4 must hold in both directions. In particular, the probabilities that X and Y are in node \hat{S} at time t are identical. One simply needs to add the probabilities of all directed paths of length t which end in \hat{S} (provided that such paths exist), where the probability of a path is the product of the μ_{ab} along the path. Therefore,

X and Y terminate with equal probability, and on average within the same number of time steps. \square

Some remarks to clarify the usefulness of this result: First of all, in contrast to Theorems 4 to 6, Theorem 7 does not explicitly require Θ_j^* to be uncritical. In fact, this is implicit in the definition of probabilistic bisimulation. Moreover, while the other theorems relate Y and X for identical histories H^t , Theorem 7 relates Y and X for *related* histories H_Y^t and H_X^t , making it more generally applicable. Finally, Theorem 7 has an important practical implication: it tells us that we can use efficient methods for model checking (e.g. [Baier, 1996, Larsen and Skou, 1991]) to verify optimality of Θ_j^* . In fact, it can be shown that for Property 4 to hold (albeit not in the other direction) it suffices that Y be a *probabilistic simulation* [Baier, 1996] of X , which is a coarser preorder than probabilistic bisimulation. However, algorithms for checking probabilistic simulation (e.g. [Baier, 1996]) are computationally much more expensive (and fewer) than those for probabilistic bisimulation, hence their practical use is currently limited.

6 CONCLUSION

This paper complements works such as [Albrecht and Ramamoorthy, 2013, Barrett et al., 2011, Carmel and Markovitch, 1999] — with a focus on HBA due to its generality — by providing answers to two important theoretical questions: “Under what conditions does HBA learn the type distribution of the game?” and “How accurate must the user-defined type spaces be for HBA to solve its task?” With respect to the first question, we analyse the convergence properties of two existing posteriors and propose a new posterior which can learn correlated type distributions. This provides the user with formal reasons as to which posterior should be chosen for the problem at hand. With respect to the second question, we describe a methodology in which we analyse the requirements of several termination guarantees, and we provide a novel characterisation of optimality which is based on the notion of probabilistic bisimulation. This gives the user a formal yet practically useful criterion of what constitutes optimal type spaces. The results of this work improve our understanding of how a method such as HBA can be used to effectively solve agent interaction problems in which the behaviour of other agents is not a priori known.

There are several interesting directions for future work. For instance, it is unclear what effect the prior probabilities P_j have on the performance of HBA, and if a criterion for optimal P_j could be derived. Furthermore, since our convergence proofs in Section 4 are asymptotic, it would be interesting to know if useful finite-time error bounds exist. Finally, our analysis in Section 5 is general in the sense that it applies to any posterior. This could be refined by an analysis which commits to a specific posterior.

References

- S. Albrecht and S. Ramamoorthy. Comparative evaluation of MAL algorithms in a diverse set of ad hoc team problems. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, 2012.
- S. Albrecht and S. Ramamoorthy. A game-theoretic model and best-response learning method for ad hoc coordination in multiagent systems. In *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems*, 2013.
- D. Avrahami-Zilberbrand and G. Kaminka. Incorporating observer biases in keyhole plan recognition (efficiently!). In *Proceedings of the 22nd Conference on Artificial Intelligence*, 2007.
- C. Baier. Polynomial time algorithms for testing probabilistic bisimulation and simulation. In *Proceedings of the 8th International Conference on Computer Aided Verification, Lecture Notes in Computer Science*, volume 1102, pages 38–49. Springer, 1996.
- S. Barrett, P. Stone, and S. Kraus. Empirical evaluation of ad hoc teamwork in the pursuit domain. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems*, 2011.
- E. Bonchek-Dokow, G. Kaminka, and C. Domshlak. Distinguishing between intentional and unintentional sequences of actions. In *Proceedings of the 9th International Conference on Cognitive Modeling*, 2009.
- M. Bowling and P. McCracken. Coordination and adaptation in impromptu teams. In *Proceedings of the 20th National Conference on Artificial Intelligence*, 2005.
- S. Carberry. Techniques for plan recognition. *User Modeling and User-Adapted Interaction*, 11(1-2):31–48, 2001.
- D. Carmel and S. Markovitch. Exploration strategies for model-based learning in multi-agent systems: Exploration strategies. *Autonomous Agents and Multi-Agent Systems*, 2(2):141–172, 1999.
- P. Doshi, Y. Zeng, and Q. Chen. Graphical models for interactive POMDPs: representations and solutions. *Autonomous Agents and Multi-Agent Systems*, 18(3):376–416, 2009.
- I. Gilboa and D. Schmeidler. *A theory of case-based decisions*. Cambridge University Press, 2001.
- P. Gmytrasiewicz and P. Doshi. A framework for sequential planning in multiagent settings. *Journal of Artificial Intelligence Research*, 24(1):49–79, 2005.
- P. Gmytrasiewicz and E. Durfee. Rational coordination in multi-agent environments. *Autonomous Agents and Multi-Agent Systems*, 3(4):319–350, 2000.
- H. Hansson and B. Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6(5): 512–535, 1994.
- E. Kalai and E. Lehrer. Rational learning leads to Nash equilibrium. *Econometrica*, pages 1019–1045, 1993.
- K. Larsen and A. Skou. Bisimulation through probabilistic testing. *Information and Computation*, 94(1):1–28, 1991.
- P. Stone, G. Kaminka, S. Kraus, and J. Rosenschein. Ad hoc autonomous agent teams: Collaboration without pre-coordination. In *Proceedings of the 24th Conference on Artificial Intelligence*, 2010.
- R. Sutton and A. Barto. *Reinforcement learning: An introduction*. The MIT Press, 1998.

Accelerating MCMC via Parallel Predictive Prefetching

Elaine Angelino, Eddie Kohler, Amos Waterland, Margo Seltzer and Ryan P. Adams

School of Engineering and Applied Sciences

Harvard University

{elaine,margo}@eecs.harvard.edu {kohler,apw,rpa}@seas.harvard.edu

Abstract

Parallel predictive prefetching is a new framework for accelerating a large class of widely-used Markov chain Monte Carlo (MCMC) algorithms. It speculatively evaluates many potential steps of an MCMC chain in parallel while exploiting fast, iterative approximations to the target density. This can accelerate sampling from target distributions in Bayesian inference problems. Our approach takes advantage of whatever parallel resources are available, but produces results *exactly equivalent* to standard serial execution. In the initial burn-in phase of chain evaluation, we achieve speedup close to linear in the number of available cores.

1 INTRODUCTION

Probabilistic modeling is one of the mainstays of modern machine learning, and Bayesian methods are particularly appealing due to their ability to represent uncertainty in parameter estimates and latent variables. Unfortunately, Bayesian inference can be difficult in the real world. Many problems are not amenable to exact inference, and so require approximate inference in the form of Monte Carlo estimates or variational approximations. These procedures require many evaluations of a target posterior density, and each evaluation can be expensive, especially on large data sets. Our work accelerates Markov chain Monte Carlo (MCMC) but, in contrast to other recent proposals, we arrive at a method in which the stationary distribution is *exactly* the target posterior. This method exploits approximations to the target density to speculatively evaluate many potential future steps of the chain in parallel.

The increasing availability of multi-core machines, and many-core cluster deployments, led to our focus on parallelism. Unfortunately, the execution of MCMC algorithms such as Metropolis-Hastings (MH) is inherently serial. One

can run many independent chains at once, but this does not change the mixing time for any single chain. Since mixing time can be prohibitively large, especially when the target function is high-dimensional and multi-modal, this embarrassingly parallel approach tends not to reduce the time to achieve a useful estimator. Sometimes the target function evaluation can be parallelized, or multiple chains in an ensemble method can be run in parallel, but these strategies are not available in general.

We instead use speculative execution to parallelize a large class of MCMC methods. This approach, sometimes called *prefetching*, has received some attention in the past decade, but does not seem to be widely recognized. Speculative execution is the general technique of optimistically performing computational work that might be eventually useful. To understand speculative execution in the context of MCMC, consider the MH algorithm in Algorithm 1, in which each iteration consists of a proposal that is stochastically accepted or rejected (Metropolis et al., 1953). MH uses randomness in two ways: to generate uniform random variables and to generate proposals. Given a random stream and an initial state, all possible future states of the chain can be thought of as the nodes of a binary tree (Figure 1). Serial execution of MH yields a sequence of states that maps to a single path of nodes through the tree. Starting at the root, each transition stochastically chooses between the current state (left child) and the proposal (right child). This requires evaluating the target density at the root and each subsequent proposal. The main goal of prefetching is to perform these evaluations in parallel. However, only the immediate transition that compares the root of the tree to the first proposal is known *a priori* to be on the true computational path. Prefetching schemes use parallel cores to evaluate these two nodes and also speculatively evaluate additional nodes further down the tree.

An effective prefetching implementation must overcome several challenges. Some involve correctness; for example, care is required in the treatment of pseudo-randomness lest bias be introduced (*i.e.*, each node’s source of randomness must produce exactly the same results as it would in a serial

Algorithm 1 Metropolis-Hastings

Input: initial state θ_0 , number of iterations T , target $\pi(\theta)$, proposal $q(\theta' | \theta)$
Output: samples $\theta_1, \dots, \theta_T$
for $t = 0, \dots, T - 1$ **do**
 $\theta' \sim q(\theta' | \theta_t)$
 $u \sim \text{Unif}(0, 1)$
 if $\frac{\pi(\theta')q(\theta_t | \theta')}{\pi(\theta_t)q(\theta' | \theta_t)} > u$ **then**
 $\theta_{t+1} = \theta'$
 else
 $\theta_{t+1} = \theta_t$
 end if
end for

execution). But the key challenge is performance. A naïve scheduling scheme always requires $\approx 2^s$ parallel cores to achieve a speedup of s . Less naïve schemes improve on this speedup using the average proposal acceptance rate: if most proposals are rejected, a prefetching implementation should prefetch more heavily among the right children of the left-most branch, *i.e.*, the path representing a sequence of rejected proposals. Although in practice the optimal acceptance rate is less than 0.5 (Gelman et al., 1996), tiny acceptance rates, which lead to good speedup, cause less effective mixing. If the acceptance rate is set near the 0.234 value of Gelman et al., speedup is still at most logarithmic.

We evaluate a new scheduling approach that uses local information to improve speedup relative to other prefetching schemes. We adaptively adjust speculation based not only on the local average proposal acceptance rate (which changes as evaluation progresses), but also on the actual random deviate used at each state. Even better, we make use of any available fast approximations to the transition operator. Though these approximations are not required, when they are available or learnable, we leverage them to make better scheduling decisions.

We present results using a series of increasingly expensive but more accurate approximations. These decisions are further improved by modeling the error of these approximations, and thus the uncertainty of the scheduling decisions. Performance depends critically on how we model the approximations, and a key insight is in our error model for this setting; much smaller error, and therefore more precise prediction, is obtained by modeling the error of the *difference* between two proposal evaluations, rather than evaluating the errors of the proposals separately. Our current implementation uses approximations that correspond to incremental evaluation of the target distribution, but our framework does not require this. We could use other examples of target density approximation, including exploiting closed form approximations such as Taylor series (Christen and Fox, 2005) and fitting linear or Gaussian Process regressions (Conrad et al., 2014).

Motivated by large-scale Bayesian inference, we present results using incremental approximations that arise from evaluating a subset of factors in a larger product. As we show on inference problems using both real and synthetic data, our system takes advantage of parallelism to speed up the wall-clock time of serial Markov chain evaluation. Unlike prior systems, we achieve near-linear speedup during burn-in on up to 64 cores spread across two or more machines. As evaluation progresses, speedup eventually decreases to logarithmic in the number of cores; we show why this is hard to avoid.

2 RELATED WORK

In this section, we summarize existing parallel strategies for accelerating MCMC, motivated by the computational cost of MCMC. This cost is most often determined by evaluation of the target density relative to mixing. In Metropolis-Hastings, it is incurred when the target is evaluated to determine the acceptance ratio of a proposed move; in slice sampling (Neal, 2003) an expensive target slows both bracket expansion and contraction. We focus on the increasingly common case where the target is expensive and the dominant computational cost. This evaluation can sometimes be parallelized directly, *e.g.*, when the target function is a product of many individually expensive terms. This can arise in Bayesian inference if the target easily decomposes into one likelihood term for each data item. Practically achievable speedup in this setting is limited by the communication and computational costs associated with aggregating the partial evaluations. In general, the target function cannot be parallelized; we divide methods that accelerate MCMC via other sources of parallelism into two classes: ensemble sampling and prefetching.

Other work speeds up MCMC evaluation using approximation. Stochastic variational inference techniques achieve scalable approximate inference via randomized approximations of gradients (Hoffman et al., 2013), while recent developments in MCMC have implemented efficient transition operators that lead to approximate stationary distributions (Welling and Teh, 2011; Korattikara et al., 2014; Bardenet et al., 2014). Recent other work uses a lower bound on the local likelihood factor to simulate from the exact posterior distribution while evaluating only a subset of the data at each iteration (Maclaurin and Adams, 2014). Unlike such prior work, we speed up *exact* evaluation of many existing MCMC algorithms.

2.1 ENSEMBLE SAMPLERS

Ensemble (or *population*) methods for sampling run multiple chains and accelerate mixing by sharing information between the chains. The individual chains can be simu-

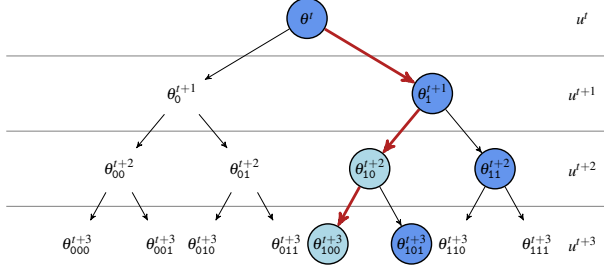


Figure 1: Schematic of a MH simulation superimposed on the binary tree of all possible chains. Each level of the tree represents an iteration, where branching to the right/left indicates accepting/rejecting a proposal. Random variates (on right) are shared across each layer. Thick red arrows highlight one simulated chain starting at the root θ^t ; the first proposal is accepted and the next two are rejected, yielding as output: $\theta_1^{t+1}, \theta_{10}^{t+2}, \theta_{100}^{t+3}$. Dark filled circles indicate states where the target density is evaluated during simulation. Those not on the chain’s path correspond to rejected proposals. Their siblings are pale filled circles on this path; since each is a copy of its parent, the target density does not need to be reevaluated to compute the next transition.

lated in parallel; any information sharing between chains requires communication. Examples include parallel tempering (Swendsen and Wang, 1986), the emcee implementation (Foreman-Mackey et al., 2012) of affine-invariant ensemble sampling (Goodman and Weare, 2010), and a parallel implementation of generalized elliptical slice sampling (Nishihara et al., 2014).

2.2 PREFETCHING

The second class of parallel MCMC algorithms uses parallelism through speculative execution to accelerate individual chains. This idea is called *prefetching* in some of the literature. To the best of our knowledge, prefetching has only been studied in the context of MH-style algorithms where, at each iteration, a single new proposal is drawn from a proposal distribution and stochastically accepted or rejected. The typical body of an MH implementation is a loop containing a single conditional statement and two associated branches. One can then view the possible execution paths as a binary tree, as illustrated in Figure 1. The vanilla version of prefetching speculatively evaluates all paths in this binary tree (Brockwell, 2006). The correct path will be exactly one of these, so with J cores, this approach achieves a speedup of $\log_2 J$ with respect to single core execution, ignoring communication and bookkeeping overheads.

Naïve prefetching can be improved by observing that the two branches are not taken with equal probability. On average, the left-most branch, corresponding to a sequence of rejected proposals, tends to be more probable; the classic

result for the optimal MH acceptance rate is 0.234 (Roberts et al., 1997), so most prefetching scheduling policies have been built around the expectation of rejection. Let $\alpha \leq 0.5$ be the expected probability of accepting a proposal. Byrd et al. (2008) introduced a procedure, called “speculative moves,” that speculatively evaluates only along the “reject” branch of the binary tree; in Figure 1, this corresponds to the left-most branch. In each round of their algorithm, only the first k out of $J - 1$ extra cores perform useful work, where k is the number of rejected proposals before the first accepted proposal, starting from the root of the tree. The expected speedup is then:

$$1 + \mathbb{E}(k) < 1 + \sum_{k=0}^{\infty} k(1 - \alpha)^k \alpha < 1 + \frac{1 - \alpha}{\alpha} = \frac{1}{\alpha}.$$

Note that the first term on the left is due to the core at the root of the tree, which always performs useful computation in the prefetching scheme. When $\alpha = 0.23$, this scheme yields a maximum expected speedup of about 4.3; it achieves an expected speedup of about 4 with 16 cores. If only a few cores are available, this may be a reasonable policy, but if many cores are available, their work is essentially wasted. In contrast, the naïve prefetching policy achieves speedup that grows as the log of the number of cores. Byrd et al. (2010) later considered the special case where the evaluation of the likelihood function occurs on two timescales, slow and fast. They call this method “speculative chains”; it modifies “speculative moves” so that whenever the evaluation of the likelihood function is slow, any available cores are used to speculatively evaluate the subsequent chain, assuming the slow step resulted in an accept.

In work closely related to ours, Strid (2010) extend the naïve prefetching scheme to allocate cores according to the optimal “tree shape” with respect to various assumptions about the probability of rejecting a proposal, *i.e.*, by greedily allocating cores to nodes that maximize the depth of speculative computation expected to be correct (Strid, 2010). Their static prefetching scheme assumes a fixed acceptance rate; versions of this were proposed earlier in the context of simulated annealing (Witte et al., 1991). Their dynamic scheme estimates acceptance probabilities, *e.g.*, at each level of the tree by drawing empirical MH samples (100,000 in the evaluation), or at each branch in the tree by computing $\min\{\beta, r\}$ where β is a constant ($\beta = 1$ in the evaluation) and r is an estimate of the MH ratio based on a fast approximation to the target function. Alternatively, Strid proposes using the approximate target function to identify the single most likely path on which to perform speculative computation. Strid also combines prefetching with other sources of parallelism to obtain a multiplicative effect. To the best of our knowledge, these methods have been developed for MH algorithms and evaluated on up to 64 cores, although usually many fewer.

3 PREDICTIVE PREFETCHING

We propose *predictive prefetching*, an improved scheduling approach that accelerates exact MCMC. Like Strid’s dynamic prefetching procedure, we also exploit inexpensive but approximate target evaluations. However, there are several fundamental differences between our approach and existing prefetching methods. We combine approximate target evaluation with the fact that the random stream used by a MCMC algorithm can be generated in advance and thus incorporated into the estimates of the acceptance probabilities at each branch in the binary tree. Critically, we also model the error of the target density approximation, and thus the uncertainty of whether a proposal will be accepted. In addition, we identify a broad class of MCMC algorithms that could benefit from prefetching, not just MH, and we show how prefetching can exploit a series of approximations, not just a single one.

3.1 MATHEMATICAL SETUP

Consider a transition operator $T(\theta \rightarrow \theta')$ which has $\pi(\theta)$ as its stationary distribution on state space Θ . Simulation of such an operator typically proceeds using an “external” source of pseudo-random numbers that can, without loss of generality, be assumed to be drawn uniformly on the unit hypercube \mathcal{U} . The transition operator is then a deterministic function $T : \Theta \times \mathcal{U} \rightarrow \Theta$. Most practical transition operators – Metropolis–Hastings, slice sampling, *etc.* – are actually compositions of two such functions, however. The first function produces a countable set of candidate points in Θ , here denoted $Q : \Theta \times \mathcal{U}_Q \rightarrow \mathcal{P}(\Theta)$, where $\mathcal{P}(\Theta)$ is the power set of Θ . The second function $R : \mathcal{P}(\Theta) \times \mathcal{U}_R \rightarrow \Theta$ then chooses one of the candidates for the next state in the Markov chain. Here we have used \mathcal{U}_Q and \mathcal{U}_R to indicate the orthogonal parts of \mathcal{U} relevant to each part of the operator. In this setup, the basic Metropolis–Hastings algorithm uses $Q(\cdot)$ to produce a tuple of the current point and a proposed point, while multiple-try MH (Liu et al., 2000) and delayed-rejection MH (Tierney and Mira, 1999; Green and Mira, 2001) create a larger set that includes the current point. In the exponential-shrinkage variant of slice sampling (Neal, 2003), $Q(\cdot)$ produces an infinite sequence of candidates that converges to, but does not include, the current point.

This setup is a somewhat more elaborate treatment than usual, but this is intended to serve two purposes: 1) make it clear that there is a separation between generating a set of possible candidates via $Q(\cdot)$ and selecting among them with $R(\cdot)$, and 2) highlight that both of these functions are deterministic functions, given the pseudo-random variates. Others have pointed out this “deterministic given the randomness” view, and used it to construct alternative approaches to MCMC (Propp and Wilson, 1996; Neal, 2012).

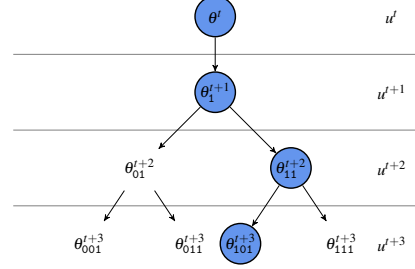


Figure 2: Schematic of the same MH simulation as in Figure 1, this time superimposed on the jobtree. This tree includes only those nodes in the original MH tree where a new state is introduced and thus the target density must be evaluated when comparing such a state to another. The filled circles, corresponding to states where the target density is evaluated in a serial MH execution, are now directly connected by a single path.

We separately consider $Q(\cdot)$ and $R(\cdot)$ because it is generally the case that $Q(\cdot)$ is inexpensive to evaluate and does not require computation of the target density $\pi(\theta)$, while $R(\cdot)$ must compare the target density at the candidate locations and so represents the bulk of the computational burden. Parallel predictive prefetching observes that, since $Q(\cdot)$ is cheap and the pseudo-random variates can be produced in any order, the tree of possible future states of the Markov chain can be constructed before any of the $R(\cdot)$ functions are evaluated, as in Figure 1. The sequence of $R(\cdot)$ evaluations simply chooses a path down this tree. Parallelism can be achieved by speculatively choosing to evaluate $R(\{\theta_i\}, u)$ for some part of the tree that has not yet been reached. If this node in the tree is eventually reached, then we achieve a speedup.

For clarity, we henceforth focus on the straightforward random-walk Metropolis–Hastings operator. In this special case, $Q(\cdot)$ produces a tuple of the current point and a proposal, and the function $R : \Theta \times \Theta \times (0, 1) \rightarrow \Theta$ takes these two points, along with a uniform random variate u in $(0, 1)$, and selects one of the two inputs via:

$$R(\theta, \theta', u) = \begin{cases} \theta' & \text{if } u \frac{q(\theta'|\theta)}{q(\theta|\theta')} < \frac{\pi(\theta')}{\pi(\theta)}, \\ \theta & \text{otherwise} \end{cases}, \quad (1)$$

where $q(\cdot|\cdot)$ is the proposal density corresponding to $Q(\cdot)$. We write the acceptance ratio in this somewhat unusual fashion to highlight the fact that the left-hand side of the inequality does not require evaluation of the target density and is easy to precompute.

3.2 THE JOBTREE

While the MH state tree in Figure 1 effectively represents a simulated chain as a path, it yields an awkward

representation of the computation necessary to produce a chain. Specifically, transitions to right children (when a proposal is accepted) align with this path but transitions to left children (when a proposal is rejected) branch off it. For our prefetching framework, we wanted a better representation of this computation. To this end, we introduced the Metropolis–Hastings *jobtree*, depicted in Figure 2. It contains the same information as the MH state tree but represents only those states where new computation occurs, *i.e.*, where the target density must be evaluated in order to compare such a state to another. Like the original tree, the jobtree is generally binary, except that the root has only one child. It includes the root node and all right children of the MH state tree, corresponding to the current state and all possible subsequent proposals – together, these specify the possible distinct states and at what iteration each would first appear. Paths on the jobtree represent computation in the sense that they map to sequences of states where the target density is evaluated during serial MH simulation; we call any such path a *computation path*.

3.3 EXPLOITING PREDICTIONS

Consider a prefetching framework with J cores that uses one core to compute the immediate transition and the others to precompute transitions for possible future iterations. If each precomputation falls along the actual Markov chain, the framework will achieve the ideal linear speedup proportional to J . If some of them do not fall along the chain, the framework will fail to scale perfectly with the available resources. For instance, recall that the naïve framework that evaluates transitions based on breadth-first search of the prefetching state tree (Figure 1) will achieve speedup proportional to $\log_2 J$. Good speedup thus depends on making good predictions of what path will be taken on the tree, which is in turn determined by our prediction of whether the threshold will be exceeded in Eq. 1.

Let ρ denote a node on the tree, θ_ρ indicate the current state at ρ , and θ'_ρ indicate the proposal. We define

$$r_\rho = u_\rho \frac{q(\theta'_\rho | \theta_\rho)}{q(\theta_\rho | \theta'_\rho)} \quad (2)$$

where u_ρ is the MH threshold variate for node ρ . The Markov chain’s steps are determined by iterations of computing the indicator function $\iota_\rho = \mathbb{I}(r_\rho < \pi(\theta'_\rho)/\pi(\theta_\rho))$, where a proposal is accepted iff $\iota_\rho = 1$. The quantities θ_ρ , θ'_ρ , and r_ρ can be inexpensively computed at any time from the stream of pseudo-random numbers, without examining the expensive target $\pi(\cdot)$. The random variate u_ρ depends only on the depth (iteration) of ρ .

The precomputation schedule should maximize expected speedup, which corresponds to the expected number of precomputations along the true computation path in the job-

tree. To maximize this quantity, the framework needs to anticipate which branches of the jobtree are likely to be taken. The root node and its only child are always evaluated. We associate with each remaining node ρ in the jobtree a predictor ψ_ρ that models the probability that the proposal is accepted, given approximate or partial information. For example, suppose that $\tilde{\pi}(\cdot)$ is an approximation to the target density $\pi(\cdot)$, and assume that in log space, the error of this approximation relative to the target density is normally distributed with some variance σ^2 . Then, we could write the predictor as:

$$\begin{aligned} \psi_\rho &= \Pr\left(\log r_\rho < \log \pi(\theta'_\rho) - \log \pi(\theta_\rho) \mid \tilde{\pi}(\cdot), \sigma^2\right) \quad (3) \\ &= \int_{\log r_\rho}^{\infty} \mathcal{N}\left(z \mid \log \tilde{\pi}(\theta'_\rho) - \log \tilde{\pi}(\theta_\rho), \sigma^2\right) dz. \quad (4) \end{aligned}$$

As more information becomes available in the form of better approximators, the predictor ψ_ρ will change. When $\tilde{\pi}(\cdot) = \pi(\cdot)$, the predictor equals the indicator ι_ρ . We label the edges in the jobtree with *branch probabilities*: the edge from a node ρ to its right child has branch probability equal to the predictor ψ_ρ and the edge to its left child has branch probability $1 - \psi_\rho$. Assuming that the predictions at each node are independent, the probability that a node’s computation is used is the product of the branch probabilities along the path connecting the root to ρ ; we call this quantity the node’s *expected utility*. Those nodes with maximum expected utility should be scheduled for precomputation. (The immediate transition will always be chosen: it has no ancestors and utility 1.)

A predictor is always available – for instance, one can use the recent acceptance probability – but many problems can improve predictions using computation. To model this, we define a sequence of predictors

$$\psi_\rho^{(m)} \approx \psi_\rho, \quad m = 0, 1, 2, \dots, N, \quad (5)$$

where increasing m implies increasing accuracy, and $\psi_\rho^{(N)} = \iota_\rho$. Workers move through this sequence until they perform the exact computation. The predictor sequence affects scheduling decisions: once it becomes sufficiently certain that a worker’s branch will not be taken, that worker and its descendants should be reallocated to more promising branches. Ultimately, every true step of the Markov chain is computed to completion. The approach simulates from the true stationary distribution, not an approximation thereof. The estimators are used only in prefetching.

There are several schemes for producing this estimator sequence, and predictive prefetching applies to any Markov chain Monte Carlo problem for which approximations are available. We focus on the important case where improved estimators are obtained by including more and more of the data in the posterior target distribution.

3.4 LARGE-SCALE BAYESIAN INFERENCE

In Bayesian inference with MCMC, the target density is a (possibly unnormalized) posterior distribution. In most modeling problems, such as those using graphical models, the target density can be decomposed into a product of terms. If the data are conditionally independent given the model parameters, there is a factor for each of the N data:

$$\pi(\theta | \mathbf{x}) \propto \pi_0(\theta) \pi(\mathbf{x} | \theta) = \pi_0(\theta) \prod_{n=1}^N \pi(x_n | \theta). \quad (6)$$

Here $\pi_0(\theta)$ is a prior distribution and $\pi(x_n | \theta)$ is the likelihood term associated with the n th datum. The logarithm of the target distribution is a sum of terms

$$\mathcal{L}(\theta) = \log \pi(\theta | \mathbf{x}) = \log \pi_0(\theta) + \sum_{n=1}^N \log \pi(x_n | \theta) + c,$$

where c is an unknown constant that does not depend on θ and can be ignored. Our predictive prefetching algorithm uses this to form predictors ψ_ρ as in Eq. 3; we again reframe ψ_ρ using log probabilities as

$$\psi_\rho \approx \Pr(\log r_\rho < \mathcal{L}(\theta') - \mathcal{L}(\theta)), \quad (7)$$

where r_ρ is the precomputed random MH threshold of Eq. 2. One approach to forming this predictor is to use a normal model for each $\mathcal{L}(\theta)$, as in Korattikara et al. (2014). However, we can achieve a better estimator with lower variance by modeling $\mathcal{L}(\theta)$ and $\mathcal{L}(\theta')$ together, rather than separately. Expanding each log likelihood gives:

$$\mathcal{L}(\theta') - \mathcal{L}(\theta) = \log \pi_0(\theta') - \log \pi_0(\theta) + \sum_{n=1}^N \Delta_n \quad (8)$$

$$\Delta_n = \log \pi(x_n | \theta') - \log \pi(x_n | \theta). \quad (9)$$

In Bayesian posterior sampling, the proposal θ' is usually a perturbation of θ and so we expect $\log \pi(x_n | \theta')$ to be correlated with $\log \pi(x_n | \theta)$. In this case, the differences Δ_n occur on a smaller scale and have a smaller variance compared to the variance due to $\log \pi(x_n | \theta)$ across data terms.

A concrete sequence of estimators is obtained by subsampling the data. Let $\{\Delta_n\}_{n=1}^m$ be a subsample of size $m < N$, without replacement, from $\{\Delta_n\}_{n=1}^N$. This subsample can be used to construct an unbiased estimate of $\mathcal{L}(\theta') - \mathcal{L}(\theta)$. We model the terms of this subsample as i.i.d. from a normal distribution with bounded variance σ^2 , leading to:

$$\mathcal{L}(\theta') - \mathcal{L}(\theta) \sim \mathcal{N}(\hat{\mu}_m, \hat{\sigma}_m^2). \quad (10)$$

The mean estimate $\hat{\mu}_m$ is empirically computable:

$$\hat{\mu}_m = \log \pi_0(\theta') - \log \pi_0(\theta) + \frac{N}{m} \sum_{n=1}^m \Delta_n. \quad (11)$$

The error estimate $\hat{\sigma}_m$ may be derived from s_m/\sqrt{m} , where s_m is the empirical standard deviation of the m sub-sampled Δ_n terms. To obtain a confidence interval for the sum of N terms, we multiply this estimate by N and the finite population correction $\sqrt{(N-m)/N}$, giving:

$$\hat{\sigma}_m = s_m \sqrt{\frac{N(N-m)}{m}}. \quad (12)$$

We can now form the predictor $\psi_\rho^{(m)}$ by considering the tail probability for $\log r_\rho$:

$$\psi_\rho^{(m)} = \int_{\log r_\rho}^{\infty} \mathcal{N}(z | \hat{\mu}_m, \hat{\sigma}_m^2) dz \quad (13)$$

$$= \frac{1}{2} \left[1 + \operatorname{erf} \left(\frac{\log \hat{\mu}_m - \log r_\rho}{\sqrt{2} \hat{\sigma}_m} \right) \right]. \quad (14)$$

3.5 SYSTEM

Our system is fully parallel and runs on network clusters of computers, each of which may comprise multiple cores. We do not perform any affinity scheduling, so all cores are treated identically whether they co-reside on the same machine or not. Our system does not use shared memory; rather, cores communicate via message passing. Note that we assign one thread to each core. To date, the largest installation on which we have run is a shared cluster of 5 machines with a total of 160 cores, on which we have used in parallel at least 64 cores spanning a minimum of 2 machines.

Our system executes predictive prefetching as follows. A master node manages the jobtree and distributes a different node in the jobtree to each worker. When a worker receives a message to compute on node ρ , it first computes the corresponding proposal θ_ρ (which may consume values from the random sequence). It asynchronously transmits the proposal and the new point in the random sequence back to the master. It then starts evaluating the target density, producing progressively improved approximations to the target that it periodically reports back to the master. Meanwhile, the master uses estimates of $\mathcal{L}(\theta'_\rho) - \mathcal{L}(\theta_\rho)$ values, the appropriate r_ρ constants, and an adaptive estimate of the current acceptance probability to calculate the predictor $\psi_\rho^{(m)}$ for each node in the evaluation tree. To assign a worker to a node, the master stochastically traverses down the jobtree from the root, following branches according to their branch probabilities, until it finds a node that is inactive, *i.e.*, no other worker is currently working on it. In this way, the master stochastically assigns workers to those nodes with highest expected utility. During computation, expected utilities change. When the master notices that the expected utility of a worker's node falls below that of other inactive nodes, the master tells the worker to abandon its work. If the abandoned proposal becomes likely again, a worker will pick it up where the earlier worker left off.

J	Burn-in					
	$i_1 = 9575$		$i_2 = 24000$		$i_3 = 50000$	
1	16674	—	41978	—	87500	—
16	2730	$6.1\times$	8678	$4.3\times$	20318	$4.3\times$
32	1731	$9.6\times$	7539	$5.6\times$	19046	$4.6\times$
64	989	$16.8\times$	5894	$7.1\times$	15146	$5.8\times$

Table 1: Cumulative time (in seconds) and speedup for evaluating the Gaussian mixture model with different numbers of workers J .

In our implementation, the target posteriors $\log \pi(\theta | \mathbf{x})$ and $\log \pi(\theta' | \mathbf{x})$ are evaluated by separate workers. Our normal model for the MH ratio based on a subsample of size m depends on the empirical mean and standard deviation of the differences Δ_n , but we use an approximation to avoid the extra communication required to keep track of all these differences. The worker for θ calculates

$$G_m(\theta) = \log \pi_0(\theta) + \frac{N}{m} \sum_{n=1}^m \log \pi(x_n | \theta) \quad (15)$$

rather than the difference mean $\hat{\mu}_m$ from Eq. 11. The master can then compute $\hat{\mu}_m = G_m(\theta') - G_m(\theta)$, but the empirical standard deviation of differences, s_m in Eq. 12, must be estimated. We set

$$s_m = \sqrt{S_m(\theta)^2 + S_m(\theta')^2 - 2\tilde{c}S_m(\theta)S_m(\theta')}, \quad (16)$$

where $S_m(\theta)$ denotes the empirical standard deviation of the $m \log \pi(x_n | \theta)$ terms, and \tilde{c} approximates the correlation between $\log \pi(x_n | \theta)$ and $\log \pi(x_n | \theta')$. We empirically observe this correlation to be very high; in all experiments we set $\tilde{c} = 0.9999$. Note that this approximation only affects the quality of our speculative predictions; it does not affect the actual decision to accept or reject the proposal θ' .

Our implementation requires at least two cores, one master and one worker. Note that when there is only one worker, it is always performing useful computation for the immediate transition at the root, leaving the master with essentially nothing to do besides some bookkeeping.

4 EXPERIMENTS

Our evaluation focuses on MH for large-scale Bayesian inference using the approximations described above (though our framework can use any approximation scheme for the target distribution). Our implementation is written in C++ and Python, and uses MPI for communication between the master and worker cores.¹ We evaluate our implementation on up to 64 cores in a multicore cluster environment in which machines are connected by 10GB ethernet and each

¹<https://github.com/elaine84/fetching>

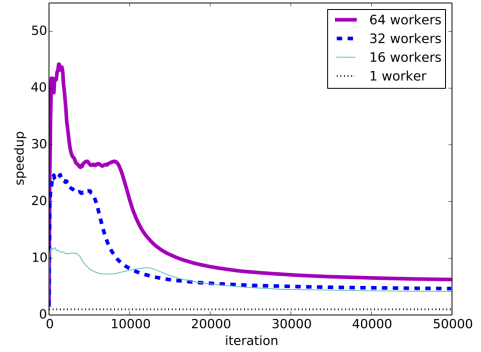


Figure 3: Cumulative speedup relative to our baseline, as a function of the number of MH iterations, for the mixture of Gaussians problem. The different curves correspond to different numbers of workers.

machine has 32 cores (four 8-core Intel Xeon E7-8837 processors). We report speedups relative to serial computation with one worker.

We evaluate our system on both synthetic and real Bayesian inference problems. First, we consider the posterior density of the eight-component mixture of eight-dimensional Gaussians used by Nishihara et al. (2014), where the likelihood involves 10^6 samples drawn from this model. Next, we consider the posterior density of a Bayesian Lasso regression (Park and Casella, 2008) that models molecular photovoltaic activity. The likelihood involves a dataset of 1.8×10^6 molecules described by 56-dimensional real-valued cheminformatic features (Olivares-Amaya et al., 2011; Amador-Bedolla et al., 2013); each response is real-valued and corresponds to a lengthy density functional theory calculation (Hachmann et al., 2011, 2014).

In our experiments, we use a spherical Gaussian for the proposal distribution. A simple adaptive scheme sets the scale of this distribution, improving convergence relative to standard MH. Our approach falls under the provably convergent adaptive algorithms studied by Andrieu and Moulines (2006); we easily incorporated them into our framework.

We expect predictive prefetching to perform best when the densities at a proposal and corresponding current point are significantly different, which is common in the initial burn-in phase of chain evaluation. In this phase, early estimates based on small subsamples effectively predict whether the proposal is accepted or rejected. When the density at the proposal is very close to that at the current point – for example, as the proposal distribution approaches the target distribution – the outcome is inherently difficult to predict; early estimates will be uncertain or even wrong. Incorrect estimates could destroy speedup (no precomputations would be useful). We hope to do better than this worst case, and to at least achieve logarithmic speedup.

	mean	standard deviation	min	max
n_{eff}	3405	7253	50	26000
\hat{R}	1.005	0.006	1.000	1.020

Table 2: Convergence statistics after burn-in (over iterations i_2 – i_3) for the Gaussian mixture model, computed over the 64 dimensions of the model.

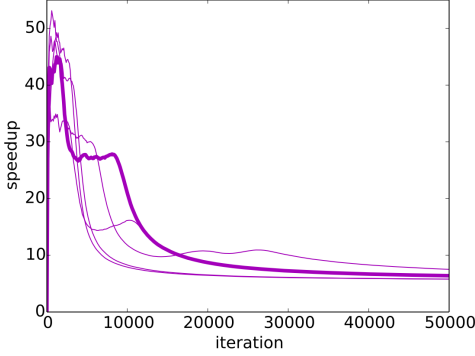


Figure 4: Cumulative speedup relative to our baseline, as a function of the number of MH iterations, for the mixture of Gaussians problem. The different curves correspond to different initial conditions; all curves are for 64 workers.

In our experiments, we divide the evaluation of the target function into 100 batches. Thus, for the mixture problem, each subsample contains 10^4 data items.

Table 1 shows the results for the Gaussian mixture model. We run the model with the same initial conditions and pseudorandom sequences with varying numbers of worker threads. All experiments produce identical chains. We evaluate the cumulative time and speedup obtained at three different iteration counts. The first, $i_1 = 9575$ iterations, are burn-in. After i_1 iterations, all dimensions of samples achieve the Gelman-Rubin statistic $\hat{R} < 1.05$, computed using two independent chains, where the first $i_1/2$ samples have been discarded (Gelman and Rubin, 1992). We then run the model further to i_3 iterations. Iterations $i_2 = 24000$ through $i_3 = 50000$ are used to compute an effective number of samples n_{eff} . (Table 2 shows convergence statistics after i_3 iterations.) The results are as we hoped. The initial burn-in phase obtains better-than-logarithmic speedup (though not perfect linear speedup). With 64 workers, the chain achieves burn-in $16.8\times$ faster than with one worker. After burn-in, efficiency drops as expected, but we still achieve logarithmic speedup (rather than sub-logarithmic). At 50000 iterations, speedup for each number of workers J rounds to $\log_2 J$.

Figure 3 explains these results by graphing cumulative speedup over the whole range of iterations. The initial

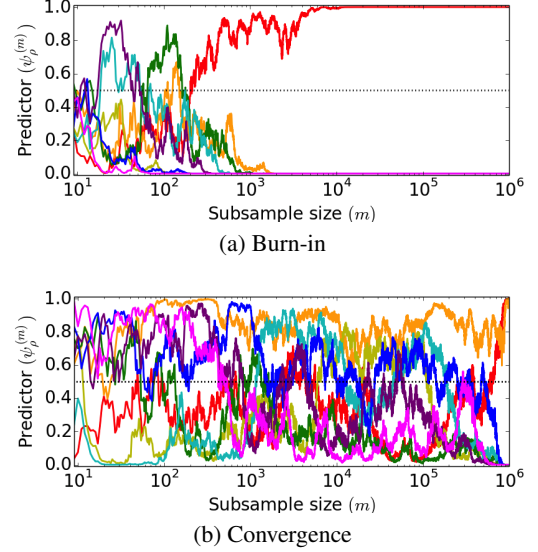


Figure 5: Example predictor trajectories for the mixture of Gaussians. We show the predictor $\psi_\rho^{(m)}$ as a function of subsample size m . Different colors indicate different proposals. Burn-in is much easier to predict than convergence.

speedup is close to linear – we briefly achieve more than $40\times$ speedup at $J = 64$ workers. As burn-in proceeds, cumulative speedup falls off to logarithmic in J . Figure 4 shows cumulative speedup for the Gaussian mixture model with several different initial conditions. We see a range of variation due to differences in the adaptive scheme during burn-in. The overall pattern is stable, however: good speedup during burn-in followed by logarithmic speedup later. Also note that speedup does not necessarily decrease steadily, or even monotonically. At some initial conditions, the chain enters an easier-to-predict region before truly burning in; while in such a region, speedup is maintained. Our system takes advantage of these regions effectively.

Figure 5 shows how our predictors behave both during and after burn-in. During burn-in, estimates are effective, and the predictor converges quite quickly to the correct indicator. After burn-in, the new proposal’s target density is close to the old proposal’s, and the estimates are similarly hard to distinguish. Sometimes the random variate r_ρ is small enough for the predictor to converge quickly to 1; more often, the predictor varies widely over time, and does not converge to 0 or 1 until almost all data are evaluated. This behavior makes logarithmic speedup a best case. Luckily, the predictor is more typically uncertain (with an intermediate value) than wrong (with an extreme value that eventually flips to the opposite value): incorrect predictors could lead to sublogarithmic speedup.

Figure 6 shows that good speedups are achievable for real problems. The speedup distribution for the Bayesian Lasso problem for molecular photovoltaic activity appears similar

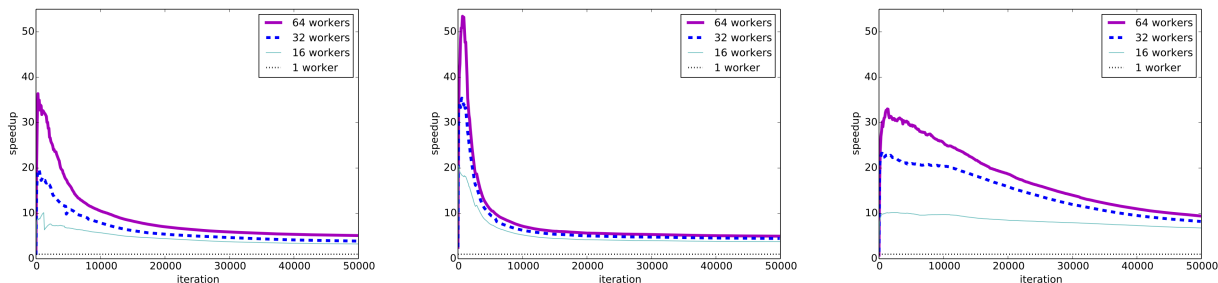


Figure 6: Cumulative speedup relative to our baseline, as a function of the number of MH iterations, for the Bayesian Lasso problem. Different curves indicate different numbers of workers. Each figure corresponds to a different initial condition.

to that of the mixture of Gaussians. There are differences, however: Lasso evaluation did not converge by 50000 iterations according to standard convergence statistics. On several initial conditions, the chain started taking small steps, and therefore dropped to logarithmic speedup, before achieving convergence. Overall performance might be improved by detecting this case and switching some speculative resources over to other initial conditions, an idea we leave for future work.

5 CONCLUSIONS

We presented parallel predictive prefetching, a general framework for accelerating many widely used MCMC algorithms that are inherently serial and often slow to converge. Our approach applies to MCMC algorithms whose transition operator can be decomposed into two functions, one that produces a countable set of candidate proposal states and a second that selects the best candidate. Predictive prefetching uses speculative computation to exploit the common setting in which (1) generating candidates is computationally fast compared to the evaluation required to select the best candidate, and (2) this evaluation can be approximated quickly. Our first focus has been on the MH algorithm, in which predictive prefetching exploits a sequence of increasingly accurate predictors for the decision to accept or reject a proposed state. Our second focus has been on large-scale Bayesian inference, for which we identified an effective predictive model that estimates the likelihood from a subset of data. The key insight is that we model the uncertainty of these predictions with respect to the difference between the likelihood of each datum evaluated at the proposal and current state. As these evaluations are highly correlated, the variance of the differences is much smaller than the variance of the states evaluated separately, leading to significantly higher confidence in our predictions. This allows us to justify more aggressive use of parallel resources, leading to greater speedup with respect to serial execution or more naïve prefetching schemes.

The best speedup that is realistically achievable for this

problem is sublinear in the number of cores but better than logarithmic, and our results achieve this. Our approach generalizes both to schemes that learn an approximation to the target density and to other MCMC algorithms with more complex structure, such as slice sampling and more sophisticated adaptive techniques.

Acknowledgments

We thank M.P. Brenner, E.D. Cubuk, V. Kanade, Z. Liu, D. Maclaurin, A.C. Miller and R. Nishihara for helpful discussions, Aspuru-Guzik, J. Hachmann and R. Olivares-Amaya for the use of the Clean Energy Project dataset and introduction to the cheminformatic feature set, and M. Tingley for the derived features used here. This work was partially funded by DARPA Young Faculty Award N66001-12-1-4219, the National Institutes of Health under Award Number 1R01LM010213-01, a Microsoft Research New Faculty Fellowship award, and Google.

REFERENCES

- C. Amador-Bedolla, R. Olivares-Amaya, J. Hachmann, and A. Aspuru-Guzik. Towards materials informatics for organic photovoltaics. In K. Rajan, editor, *Informatics for Materials Science and Engineering*. Elsevier, Amsterdam, 2013.
- C. Andrieu and E. Moulines. On the ergodicity properties of some adaptive MCMC algorithms. *The Annals of Applied Probability*, 16(3):1462–1505, 2006.
- R. Bardenet, A. Doucet, and C. Holmes. Towards scaling up Markov chain Monte Carlo: An adaptive subsampling approach. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.
- A. E. Brockwell. Parallel Markov chain Monte Carlo simulation by pre-fetching. *Journal of Computational and Graphical Statistics*, 15(1):246–261, March 2006.
- J. M. R. Byrd, S. A. Jarvis, and A. H. Bhalerao. Reducing the run-time of MCMC programs by multithreading

- on SMP architectures. In *International Symposium on Parallel and Distributed Processing*, pages 1–8, 2008.
- J. M. R. Byrd, S. A. Jarvis, and A. H. Bhalerao. On the parallelisation of MCMC by speculative chain execution. In *IPDPS Workshops*, pages 1–8, 2010.
- J. A. Christen and C. Fox. Markov chain Monte Carlo using an approximation. *Journal of Computational and Graphical Statistics*, 14(4):795–810, 2005.
- P. R. Conrad, Y. M. Marzouk, N. S. Pillai, and A. Smith. Asymptotically exact MCMC algorithms via local approximations of computationally intensive models. *ArXiv e-prints*, Feb. 2014.
- D. Foreman-Mackey, D. W. Hogg, D. Lang, and J. Goodman. emcee: The MCMC Hammer. *Publications of the Astronomical Society of the Pacific*, 125(306), 2012.
- A. Gelman and D. B. Rubin. Inference from iterative simulation using multiple sequences. *Statistical Science*, pages 457–472, 1992.
- A. Gelman, G. O. Roberts, and W. R. Gilks. Efficient Metropolis jumping rules in Bayesian statistics. *Bayesian Statistics 5*, pages 599–607, 1996.
- J. Goodman and J. Weare. Ensemble samplers with affine invariance. *Communications in Applied Mathematics and Computational Science*, 5(1):65–80, 2010.
- P. J. Green and A. Mira. Delayed rejection in reversible jump Metropolis-Hastings. *Biometrika*, 88(4):pp. 1035–1053, 2001.
- J. Hachmann, R. Olivares-Amaya, S. Atahan-Evrenk, C. Amador-Bedolla, R. S. Sánchez-Carrera, A. Gold-Parker, L. Vogt, A. M. Brockway, and A. Aspuru-Guzik. The Harvard Clean Energy Project: Large-scale computational screening and design of organic photovoltaics on the world community grid. *The Journal of Physical Chemistry Letters*, 2(17):2241–2251, 2011.
- J. Hachmann, R. Olivares-Amaya, A. Jinich, A. L. Appleton, M. A. Blood-Forsythe, L. R. Seress, C. Román-Salgado, K. Trepte, S. Atahan-Evrenk, S. Er, S. Shrestha, R. Mondal, A. Sokolov, Z. Bao, and A. Aspuru-Guzik. Lead candidates for high-performance organic photovoltaics from high-throughput quantum chemistry - the Harvard Clean Energy Project. *Energy Environ. Sci.*, 7: 698–704, 2014.
- M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(1):1303–1347, 2013.
- A. Korattikara, Y. Chen, and M. Welling. Austerity in MCMC Land: Cutting the Metropolis-Hastings Budget. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.
- J. S. Liu, F. Liang, and W. H. Wong. The multiple-try method and local optimization in Metropolis sampling. *Journal of the American Statistical Association*, 95(449): pp. 121–134, 2000.
- D. Maclaurin and R. P. Adams. Firefly Monte Carlo: Exact MCMC with subsets of data. In *30th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2014.
- N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.
- R. M. Neal. Slice sampling. *The Annals of Statistics*, 31(3):705–767, 06 2003.
- R. M. Neal. How to view an MCMC simulation as a permutation, with applications to parallel simulation and improved importance sampling. Technical Report 1201, Dept. of Statistics, University of Toronto, 2012.
- R. Nishihara, I. Murray, and R. P. Adams. Parallel MCMC with generalized elliptical slice sampling. *Journal of Machine Learning Research*, Oct. 2014.
- R. Olivares-Amaya, C. Amador-Bedolla, J. Hachmann, S. Atahan-Evrenk, R. S. Sánchez-Carrera, L. Vogt, and A. Aspuru-Guzik. Accelerated computational discovery of high-performance materials for organic photovoltaics by means of cheminformatics. *Energy Environ. Sci.*, 4: 4849–4861, 2011.
- T. Park and G. Casella. The Bayesian Lasso. *Journal of the American Statistical Association*, 103(482):681–686, 2008.
- J. G. Propp and D. B. Wilson. Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random Structures and Algorithms*, 9(1&2): 223–252, 1996.
- G. O. Roberts, A. Gelman, and W. R. Gilks. Weak convergence and optimal scaling of random walk Metropolis algorithms. *Annals of Applied Probability*, 7:110–120, 1997.
- I. Strid. Efficient parallelisation of Metropolis-Hastings algorithms using a prefetching approach. *Computational Statistics & Data Analysis*, 54(11):2814–2835, Nov. 2010.
- R. H. Swendsen and J. S. Wang. Replica Monte Carlo simulation of spin-glasses. *Physical Review Letters*, 57(21): 2607–2609, Nov. 1986.
- L. Tierney and A. Mira. Some adaptive Monte Carlo methods for Bayesian inference. *Statistics in Medicine*, 18: 2507–2515, 1999.
- M. Welling and Y. W. Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning*, 2011.
- E. Witte, R. Chamberlain, and M. Franklin. Parallel simulated annealing using speculative computation. *IEEE Transactions on Parallel and Distributed Systems*, 2(4): 483–494, 1991.

A variational approach to stable principal component pursuit

Aleksandr Aravkin

T. J. Watson Center
IBM Research
Yorktown Heights, NY

Stephen Becker

T. J. Watson Center
IBM Research
Yorktown Heights, NY

Volkan Cevher *

LIONS
EPFL
Lausanne, Switzerland

Peder Olsen

T. J. Watson Center
IBM Research
Yorktown Heights, NY

Abstract

We introduce a new convex formulation for stable principal component pursuit (SPCP) to decompose noisy signals into low-rank and sparse representations. For numerical solutions of our SPCP formulation, we first develop a convex variational framework and then accelerate it with quasi-Newton methods. We show, via synthetic and real data experiments, that our approach offers advantages over the classical SPCP formulations in scalability and practical parameter selection.

1 INTRODUCTION

Linear superposition is a useful model for many applications, including nonlinear mixing problems. Surprisingly, we can perfectly distinguish multiple elements in a given signal using convex optimization as long as they are concise and look sufficiently different from one another. Popular examples include robust principal component analysis (RPCA) where we decompose a signal into low rank and sparse components and *stable principal component pursuit (SPCP)*, where we also seek an explicit noise component within the RPCA decomposition. Applications include alignment of occluded images (Peng et al., 2012), scene triangulation (Zhang et al., 2011), model selection (Chandrasekaran et al., 2012), face recognition, and document indexing (Candès et al., 2011).

The SPCP formulation can be mathematically stated as follows. Given a noisy matrix $Y \in \mathbb{R}^{m \times n}$, we decompose it as a sum of a low-rank matrix L and a

sparse matrix S via the following convex program

$$\begin{aligned} & \underset{L, S}{\text{minimize}} \quad \|L\|_* + \lambda_{\text{sum}} \|S\|_1 \\ & \text{subject to} \quad \|L + S - Y\|_F \leq \varepsilon, \end{aligned} \quad (\text{SPCP}_{\text{sum}})$$

where the 1-norm $\|\cdot\|_1$ and nuclear norm $\|\cdot\|_*$ are given by $\|S\|_1 = \sum_{i,j} |s_{i,j}|$, $\|L\|_* = \sum_i \sigma_i(L)$, where $\sigma(L)$ is the vector of singular values of L . In (SPCP_{sum}), the parameter $\lambda_{\text{sum}} > 0$ controls the relative importance of the low-rank term L vs. the sparse term S , and the parameter ε accounts for the unknown perturbations $Y - (L + S)$ in the data not explained by L and S .

When $\varepsilon = 0$, (SPCP_{sum}) is the “robust PCA” problem as analyzed by Candès et al. (2011); Chandrasekaran et al. (2009), and it has perfect recovery guarantees under stylized incoherence assumptions. There is even theoretical guidance for selecting a minimax optimal regularization parameter λ_{sum} (Candès et al., 2011). Unfortunately, many practical problems only approximately satisfy the idealized assumptions, and hence, we typically tune RPCA via cross-validation techniques. SPCP further complicates the practical tuning due to the additional parameter ε .

To cope with practical tuning issues of SPCP, we propose the following new variant called “max-SPCP”:

$$\begin{aligned} & \underset{L, S}{\text{minimize}} \quad \max(\|L\|_*, \lambda_{\text{max}} \|S\|_1) \\ & \text{subject to} \quad \|L + S - Y\|_F \leq \varepsilon, \end{aligned} \quad (\text{SPCP}_{\text{max}})$$

where $\lambda_{\text{max}} > 0$ acts similar to λ_{sum} . Our work shows that this new formulation offers both modeling and computational advantages over (SPCP_{sum}).

Cross-validation with (SPCP_{max}) to estimate $(\lambda_{\text{max}}, \varepsilon)$ is significantly easier than estimating $(\lambda_{\text{sum}}, \varepsilon)$ in (SPCP_{sum}). For example, given an *oracle* that provides an ideal separation $Y \simeq L_{\text{oracle}} + S_{\text{oracle}}$, we can use $\varepsilon = \|L_{\text{oracle}} + S_{\text{oracle}} - Y\|_F$ in both cases. However, while we can estimate $\lambda_{\text{max}} = \|L_{\text{oracle}}\|_* / \|S_{\text{oracle}}\|_1$, it is not clear how to choose λ_{sum} from data. Such cross

*Author’s work is supported in part by the European Commission under the grants MIRG-268398 and ERC Future Proof, and by the Swiss Science Foundation under the grants SNF 200021-132548, SNF 200021-146750 and SNF CRSII2-147633.

validation can be performed on a similar dataset, or it could be obtained from a probabilistic model.

Our convex approach for solving $(\text{SPCP}_{\text{sum}})$ generalizes to other source separation problems (Baldassarre et al., 2013) beyond SPCP. Both $(\text{SPCP}_{\text{max}})$ and $(\text{SPCP}_{\text{sum}})$ are challenging to solve when the dimensions are large. We show in this paper that these problems can be solved more efficiently by solving a few (typically 5 to 10) subproblems of a different functional form. While the efficiency of the solution algorithms for $(\text{SPCP}_{\text{sum}})$ relies heavily on the efficiency of the 1-norm and nuclear norm projections, the efficiency of our solution algorithm $(\text{SPCP}_{\text{max}})$ is preserved for arbitrary norms. Moreover, $(\text{SPCP}_{\text{max}})$ allows a faster algorithm in the standard case, discussed in Section 6.

2 A PRIMER ON SPCP

The theoretical and algorithmic research on SPCP formulations (and source separation in general) is rapidly evolving. Hence, it is important to set the stage first in terms of the available formulations to highlight our contributions.

To this end, we illustrate $(\text{SPCP}_{\text{sum}})$ and $(\text{SPCP}_{\text{max}})$ via different convex formulations. Flipping the objective and the constraints in $(\text{SPCP}_{\text{max}})$ and $(\text{SPCP}_{\text{sum}})$, we obtain the following convex programs

$$\begin{aligned} & \underset{L, S}{\text{minimize}} \quad \frac{1}{2} \|L + S - Y\|_F^2 \\ & \text{s.t.} \quad \|L\|_* + \lambda_{\text{sum}} \|S\|_1 \leq \tau_{\text{sum}} \end{aligned} \quad (\text{flip-SPCP}_{\text{sum}})$$

$$\begin{aligned} & \underset{L, S}{\text{minimize}} \quad \frac{1}{2} \|L + S - Y\|_F^2 \\ & \text{s.t.} \quad \max(\|L\|_*, \lambda_{\text{max}} \|S\|_1) \leq \tau_{\text{max}} \end{aligned} \quad (\text{flip-SPCP}_{\text{max}})$$

Remark 2.1. *The solutions of $(\text{flip-SPCP}_{\text{sum}})$ and $(\text{flip-SPCP}_{\text{max}})$ are related to the solutions of $(\text{SPCP}_{\text{sum}})$ and $(\text{SPCP}_{\text{max}})$ via the Pareto frontier by Aravkin et al. (2013a, Theorem 2.1). If the constraint $\|L + S - Y\| \leq \varepsilon$ is tight at the solution, then there exist corresponding parameters $\tau_{\text{sum}}(\varepsilon)$ and $\tau_{\text{max}}(\varepsilon)$, for which the optimal value of $(\text{flip-SPCP}_{\text{sum}})$ and $(\text{flip-SPCP}_{\text{max}})$ is ε , and the corresponding optimal solutions (\bar{S}_s, \bar{L}_s) and (\bar{S}_m, \bar{L}_m) are also optimal for $(\text{SPCP}_{\text{sum}})$ and $(\text{SPCP}_{\text{max}})$.*

For completeness, we also include the Lagrangian formulation, which is covered by our new algorithm:

$$\begin{aligned} & \underset{L, S}{\text{minimize}} \quad \lambda_L \|L\|_* + \lambda_S \|S\|_1 + \frac{1}{2} \|L + S - Y\|_F^2 \\ & \quad \quad \quad (\text{Lag-SPCP}) \end{aligned} \quad 2$$

Problems $(\text{flip-SPCP}_{\text{max}})$ and $(\text{flip-SPCP}_{\text{sum}})$ can be solved using projected gradient and accelerated gradient methods. The disadvantage of some of these formulations is that it may not be clear how to tune the parameters. Surprisingly, an algorithm we propose in this paper can solve $(\text{SPCP}_{\text{max}})$ and $(\text{SPCP}_{\text{sum}})$ using a sequence of flipped problems that specifically exploits the structured relationship cited in Remark 2.1. In practice, we will see that better tuning also leads to faster algorithms, e.g., fixing ε ahead of time to an estimated ‘noise floor’ greatly reduces the amount of required computation if parameters are to be selected via cross-validation.

Finally, we note that in some cases, it is useful to change the $\|L + S - Y\|_F$ term to $\|\mathcal{A}(L + S - Y)\|_F$ where \mathcal{A} is a linear operator. For example, let Ω be a subset of the indices of a $m \times n$ matrix. We may only observe Y restricted to these entries, denoted $\mathcal{P}_\Omega(Y)$, in which case we choose $\mathcal{A} = \mathcal{P}_\Omega$. Most existing RPCA/SPCP algorithms adapt to the case $\mathcal{A} = \mathcal{P}_\Omega$ but this is due to the strong properties of the projection operator \mathcal{P}_Ω . The advantage of our approach is that it seamlessly handles arbitrary linear operators \mathcal{A} . In fact, it also generalizes to smooth misfit penalties, that are more robust than the Frobenius norm, including the Huber loss. Our results also generalize to some other penalties on S besides the 1-norm.

The paper proceeds as follows. In Section 3, we describe previous work and algorithms for SPCP and RPCA. In Section 4, we cast the relationships between pairs of problems $(\text{flip-SPCP}_{\text{sum}})$, $(\text{SPCP}_{\text{sum}})$ and $(\text{flip-SPCP}_{\text{max}})$, $(\text{SPCP}_{\text{max}})$ into a general variational framework, and highlight the product-space regularization structure that enables us solve the formulations of interest using corresponding flipped problems. We discuss computationally efficient projections as optimization workhorses in Section 5, and develop new accelerated projected quasi-Newton methods for the flipped and Lagrangian formulations in Section 6. Finally, we demonstrate the efficacy of the new solvers and the overall formulation on synthetic problems and a real cloud removal example in Section 7, and follow with conclusions in Section 8.

3 PRIOR ART

While problem $(\text{SPCP}_{\text{sum}})$ with $\varepsilon = 0$ has several solvers (e.g., it can be solved by applying the widely known Alternating Directions Method of Multipliers (ADMM)/Douglas-Rachford method (Combettes & Pesquet, 2007)), the formulation assumes the data are noise free. Unfortunately, the presence of noise we consider in this paper introduces a third term in the ADMM framework, where the algorithm is shown to

be non-convergent (Chen et al., 2013). Interestingly, there are only a handful of methods that can handle this case. Those using smoothing techniques no longer promote exactly sparse and/or exactly low-rank solutions (Aybat et al., 2013). Those using dual decomposition techniques require high iteration counts. Because each step requires a partial singular value decomposition (SVD) of a large matrix, it is critical that the methods only take a few iterations.

As a rough comparison, we start with related solvers that solve (SPCP_{sum}) for $\varepsilon = 0$. Wright et al. (2009a) solves an instance of (SPCP_{sum}) with $\varepsilon = 0$ and a 800×800 system in 8 hours. By switching to the (Lag-SPCP) formulation, Ganesh et al. (2009) uses the accelerated proximal gradient method (Beck & Teboulle, 2009) to solve a 1000×1000 matrix in under one hour. This is improved further in Lin et al. (2010) which again solves (SPCP_{sum}) with $\varepsilon = 0$ using the augmented Lagrangian and ADMM methods and solves a 1500×1500 system in about a minute. As a prelude to our results, our method can solve some systems of this size in about 10 seconds (c.f., Fig. 1).

In the case of (SPCP_{sum}) with $\varepsilon > 0$, Tao & Yuan (2011) propose the alternating splitting augmented Lagrangian method (ASALM), which exploits separability of the objective in the splitting scheme, and can solve a 1500×1500 system in about five minutes.

The partially smooth proximal gradient (PSPG) approach of Aybat et al. (2013) smooths just the nuclear norm term and then applies the well-known FISTA algorithm (Beck & Teboulle, 2009). Aybat et al. (2013) show that the proximity step can be solved efficiently in closed-form, and the dominant cost at every iteration is that of the partial SVD. They include some examples on video, lopsided matrices: 25000×300 or so, in about 1 minute). solving 1500×1500 formulations in under half a minute.

The nonsmooth adaptive Lagrangian (NSA) algorithm of Aybat & Iyengar (2014) is a variant of the ADMM for (SPCP_{sum}), and makes use of the insight of Aybat et al. (2013). The ADMM variant is interesting in that it splits the variable L , rather than the sum $L + S$ or residual $L + S - Y$. Their experiments solve a 1500×1500 synthetic problems in between 16 and 50 seconds (depending on accuracy).

Shen et al. (2014) develop a method exploiting low-rank matrix factorization scheme, maintaining $L = UV^T$. This technique has also been effectively used in practice for matrix completion (Aravkin et al., 2013b; Lee et al., 2010; Recht & Ré, 2011), but lacks a full convergence theory in either context. The method of (Shen et al., 2014) was an order of magnitude faster than ASALM, but encountered difficulties in some ex-

periments where the sparse component dominated the low rank component in some sense. Mansour & Vetro (2014) attack the (SPCP_{sum}) formulation using a factorized approach, together with alternating solves between (U, V) and S . Non-convex techniques also include hard thresholding approaches, e.g. the approach of Kyrillidis & Cevher (2014). While the factorization technique may potentially speed up some of the methods presented here, we leave this to future work, and only work with convex formulations.

4 VARIATIONAL FRAMEWORK

Both of the formulations of interest (SPCP_{sum}) and (SPCP_{max}) can be written as follows:

$$\min \phi(L, S) \quad \text{s.t.} \quad \rho(L + S - Y) \leq \varepsilon. \quad (4.1)$$

Classic formulations assume ρ to be the Frobenius norm; however, this restriction is not necessary, and we consider ρ to be smooth and convex. In particular, ρ can be taken to be the robust Huber penalty (Huber, 2004). Even more importantly, this formulation allows pre-composition of a smooth convex penalty with an arbitrary linear operator \mathcal{A} , which extends the proposed approach to a much more general class of problems. Note that a simple operator is already embedded in both formulations of interest:

$$L + S = \begin{bmatrix} I & I \end{bmatrix} \begin{bmatrix} L \\ S \end{bmatrix}. \quad (4.2)$$

Projection onto a set of observed indices Ω is also a simple linear operator that can be included in ρ . Operators may include different transforms (e.g., Fourier) applied to either L or S .

The main formulations of interest differ only in the functional $\phi(L, S)$. For (SPCP_{sum}), we have

$$\phi_{\text{sum}}(L, S) = \|L\|_* + \lambda_{\text{sum}} \|S\|_1,$$

while for (SPCP_{max}),

$$\phi_{\text{max}}(L, S) = \max(\|L\|_*, \lambda_{\text{max}} \|S\|_1).$$

The problem class (4.1) falls into the class of problems studied by van den Berg & Friedlander (2008, 2011) for $\rho(\cdot) = \|\cdot\|^2$ and by Aravkin et al. (2013a) for arbitrary convex ρ . Making use of this framework, we can define a value function

$$v(\tau) = \min_{L, S} \rho(\mathcal{A}(L, S) - Y) \quad \text{s.t.} \quad \phi(L, S) \leq \tau, \quad (4.3)$$

and use Newton’s method to find a solution to $v(\tau) = \varepsilon$. The approach is agnostic to the linear operator \mathcal{A} (it can be of the simple form (4.2); include restriction in the missing data case, etc.).

For both formulations of interest, ϕ is a norm defined on a product space $\mathbb{R}^{n \times m} \times \mathbb{R}^{n \times m}$, since we can write

$$\phi_{\text{sum}}(L, S) = \left\| \begin{bmatrix} \|L\|_* \\ \lambda_{\text{sum}} \|S\|_1 \end{bmatrix} \right\|_1, \quad (4.4)$$

$$\phi_{\text{max}}(L, S) = \left\| \begin{bmatrix} \|L\|_* \\ \lambda_{\text{max}} \|S\|_1 \end{bmatrix} \right\|_\infty. \quad (4.5)$$

In particular, both $\phi_{\text{sum}}(L, S)$ and $\phi_{\text{max}}(L, S)$ are *gauges*. For a convex set C containing the origin, the gauge $\gamma(x | C)$ is defined by

$$\gamma(x | C) = \inf_{\lambda} \{\lambda : x \in \lambda C\}. \quad (4.6)$$

For any norm $\|\cdot\|$, the set defining it as a gauge is simply the unit ball $\mathbb{B}_{\|\cdot\|} = \{x : \|x\| \leq 1\}$. We introduce gauges for two reasons. First, they are more general (a gauge is a norm only if C is bounded with nonempty interior and symmetric about the origin). For example, gauges trivially allow inclusion of non-negativity constraints. Second, definition (4.6) and the explicit set C simplify the exposition of the following results.

In order to implement Newton's method for (4.3), the optimization problem to evaluate $v(\tau)$ must be solved (fully or approximately) to obtain (\bar{L}, \bar{S}) . Then the τ parameter for the next (4.3) problem is updated via

$$\tau^{k+1} = \tau^k - \frac{v(\tau) - \tau}{v'(\tau)}. \quad (4.7)$$

Given (\bar{L}, \bar{S}) , $v'(\tau)$ can be written in closed form using (Aravkin et al., 2013a, Theorem 5.2), which simplifies to

$$v'(\tau) = -\phi^\circ(\mathcal{A}^T \nabla \rho(\mathcal{A}(\bar{L}, \bar{S}) - Y)), \quad (4.8)$$

with ϕ° denoting the polar gauge to ϕ . The polar gauge is precisely $\gamma(x | C^\circ)$, with

$$C^\circ = \{v : \langle v, x \rangle \leq 1 \quad \forall x \in C\}. \quad (4.9)$$

In the simplest case, where \mathcal{A} is given by (4.2), and ρ is the least squares penalty, the formula (4.8) becomes

$$v'(\tau) = -\phi^\circ \left(\begin{bmatrix} \bar{L} + \bar{S} - Y \\ \bar{L} + \bar{S} - Y \end{bmatrix} \right).$$

The main computational challenge in the approach outlined in (4.3)-(4.8) is to design a fast solver to evaluate $v(\tau)$. Section 6 does just this.

The key to RPCA is that the regularization functional ϕ is a gauge over the product space used to decompose Y into summands L and S . This makes it straightforward to compute polar results for both ϕ_{sum} and ϕ_{max} .

Theorem 4.1 (Max-Sum Duality for Gauges on Product Spaces). *Let γ_1 and γ_2 be gauges on \mathbb{R}^{n_1} and \mathbb{R}^{n_2} , and consider the function*

$$g(x, y) = \max\{\gamma_1(x), \gamma_2(y)\}.$$

Then g is a gauge, and its polar is given by

$$g^\circ(z_1, z_2) = \gamma_1^\circ(z_1) + \gamma_2^\circ(z_2).$$

Proof. Let C_1 and C_2 denote the canonical sets corresponding to gauges γ_1 and γ_2 . It immediately follows that g is a gauge for the set $C = C_1 \times C_2$, since

$$\begin{aligned} \inf\{\lambda \geq 0 | (x, y) \in \lambda C\} &= \inf\{\lambda | x \in \lambda C_1 \text{ and } y \in \lambda C_2\} \\ &= \max\{\gamma_1(x), \gamma_2(y)\}. \end{aligned}$$

By (Rockafellar, 1970, Corollary 15.1.2), the polar of the gauge of C is the support function of C , which is given by

$$\begin{aligned} \sup_{x \in C_1, y \in C_2} \langle (x, y), (z_1, z_2) \rangle &= \sup_{x \in C_1} \langle x, z_1 \rangle + \sup_{y \in C_2} \langle y, z_2 \rangle \\ &= \gamma_1^\circ(z_1) + \gamma_2^\circ(z_2). \end{aligned}$$

□

This theorem allows us to easily compute the polars for ϕ_{sum} and ϕ_{max} in terms of the polars of $\|\cdot\|_*$ and $\|\cdot\|_1$, which are the dual norms, the spectral norm and infinity norm, respectively.

Corollary 4.2 (Explicit variational formulae for (SPCP_{sum}) and (SPCP_{max})). *We have*

$$\begin{aligned} \phi_{\text{sum}}^\circ(Z_1, Z_2) &= \max \left\{ \|Z_1\|_2, \frac{1}{\lambda_{\text{sum}}} \|Z_2\|_\infty \right\} \\ \phi_{\text{max}}^\circ(Z_1, Z_2) &= \|Z_1\|_2 + \frac{1}{\lambda_{\text{max}}} \|Z_2\|_\infty, \end{aligned} \quad (4.10)$$

where $\|X\|_2$ denotes the spectral norm (largest eigenvalue of $X^T X$).

This result was also obtained by (van den Berg & Friedlander, 2011, Section 9), but is stated only for norms. Theorem 4.1 applies to gauges, and in particular now allows asymmetric gauges, so non-negativity constraints can be easily modeled.

We now have closed form solutions for $v'(\tau)$ in (4.8) for both formulations of interest. The remaining challenge is to design a fast solver for (4.3) for formulations (SPCP_{sum}) and (SPCP_{max}). We focus on this challenge in the remaining sections of the paper. We also discuss the advantage of (SPCP_{max}) from this computational perspective.

5 PROJECTIONS

In this section, we consider the computational issues of projecting onto the set defined by $\phi(L, S) \leq \tau$. For $\phi_{\text{max}}(L, S) = \max(\|L\|_*, \lambda_{\text{max}} \|S\|_1)$ this is straightforward since the set is just the product set of the

nuclear norm and ℓ_1 norm balls, and efficient projectors onto these are known. In particular, projecting an $m \times n$ matrix (without loss of generality let $m \leq n$) onto the nuclear norm ball takes $\mathcal{O}(m^2n)$ operations, and projecting it onto the ℓ_1 -ball can be done on $\mathcal{O}(mn)$ operations using fast median-finding algorithms (Brucker, 1984; Duchi et al., 2008).

For $\phi_{\text{sum}}(L, S) = \|L\|_* + \lambda_{\text{sum}}\|S\|_1$, the projection is no longer straightforward. Nonetheless, the following lemma shows this projection can be efficiently implemented.

Proposition 5.1. (*van den Berg & Friedlander, 2011, Section 5.2*) *Projection onto the scaled ℓ_1 -ball, that is, $\{x \in \mathbb{R}^d \mid \sum_{i=1}^d \alpha_i |x_i| \leq 1\}$ for some $\alpha_i > 0$, can be done in $\mathcal{O}(d \log(d))$ time.*

The proof of the proposition follows by noting that the solution can be written in a form depending only on a single scalar parameter, and this scalar can be found by sorting $(|x_i|/\alpha_i)$ followed by appropriate summations. We conjecture that fast median-finding ideas could reduce this to $\mathcal{O}(d)$ in theory, the same as the optimal complexity for the ℓ_1 -ball.

Armed with the above proposition, we state an important lemma below. For our purposes, we may think of S as a vector in \mathbb{R}^{mn} rather than a matrix in $\mathbb{R}^{m \times n}$.

Lemma 5.2. (*van den Berg & Friedlander, 2011, Section 9.2*) *Let $L = U\Sigma V^T$ and $\Sigma = \text{diag}(\sigma)$, and let $(S_i)_{i=1}^{mn}$ be any ordering of the elements of S . Then the projection of (L, S) onto the ϕ_{sum} ball is $(U \text{diag}(\hat{\sigma})V^T, \hat{S})$, where $(\hat{\sigma}, \hat{S})$ is the projection onto the scaled ℓ_1 -ball $\{(\sigma, S) \mid \sum_{j=1}^{\min(m,n)} |\sigma_j| + \sum_{i=1}^{mn} \lambda_{\text{sum}} |S_i| \leq 1\}$.*

Sketch of proof. We need to solve

$$\min_{\{(L', S') \mid \phi_{\text{sum}}(L', S') \leq 1\}} \frac{1}{2} \|L' - L\|_F^2 + \frac{1}{2} \|S' - S\|_F^2.$$

Alternatively, solve

$$\min_{S'} \min_{\{L' \mid \|L'\|_* \leq 1 - \lambda_{\text{sum}} \|S'\|_1\}} \frac{1}{2} \|L' - L\|_F^2 + \frac{1}{2} \|S' - S\|_F^2.$$

The inner minimization is equivalent to projecting onto the nuclear norm ball, and this is well-known to be soft-thresholding of the singular values. Since it depends only on the singular values, recombining the two minimization terms gives exactly a joint projection onto a scaled ℓ_1 -ball. \square

Remark 5.1. *All the references to the ℓ_1 -ball can be replaced by the intersection of the ℓ_1 -ball and the non-negative cone, and the projection is still efficient. As noted in Section 4, imposing non-negativity constraints*

is covered by the gauge results of Theorem 4.1 and Corollary 4.2. Therefore, both the variational and efficient computational framework can be applied to this interesting case.

6 SOLVING THE SUB-PROBLEM VIA PROJECTED QUASI-NEWTON METHODS

In order to accelerate the approach, we can use quasi-Newton (QN) methods since the objective has a simple structure.¹ The main challenge here is that for the $\|L\|_*$ term, it is tricky to deal with a weighted quadratic term (whereas for $\|S\|_1$, we can obtain a low-rank Hessian and solve it efficiently via coordinate descent).

We wish to solve (**flip-SPCP**_{max}). Let $X = (L, S)$ be the full variable, so we can write the objective function as $f(X) = \frac{1}{2} \|\mathcal{A}(X) - Y\|_F^2$. To simplify the exposition, we take $\mathcal{A} = (I, I)$ to be the $mn \times 2mn$ matrix, but the presented approach applies to general linear operators (including terms like \mathcal{P}_Ω). The matrix structure of L and S is not important here, so we can think of them as $mn \times 1$ vectors instead of $m \times n$ matrices.

The gradient is $\nabla f(X) = \mathcal{A}^T(\mathcal{A}(X) - Y)$. For convenience, we use $r(X) = \mathcal{A}(X) - Y$ and

$$\nabla f(X) = \begin{pmatrix} \nabla_L f(X) \\ \nabla_S f(X) \end{pmatrix} = \mathcal{A}^T \begin{pmatrix} r(X) \\ r(X) \end{pmatrix}, \quad r_k \equiv r(X_k).$$

The Hessian is $\mathcal{A}^T \mathcal{A} = \begin{pmatrix} I & I \\ I & I \end{pmatrix}$. We cannot simultaneously project (L, S) onto their constraints with this Hessian scaling (doing so would solve the original problem!), since the Hessian removes separability. Instead, we use (L_k, S_k) to approximate the cross-terms.

The true function is a quadratic, so the following

¹ We use “quasi-Newton” to mean an approximation to a Newton method and it should not be confused with methods like BFGS

quadratic expansion around $X_k = (L_k, S_k)$ is exact:

$$\begin{aligned}
f(L, S) &= f(X_k) + \left\langle \begin{pmatrix} \nabla_L f(X_k) \\ \nabla_S f(X_k) \end{pmatrix}, \begin{pmatrix} L - L_k \\ S - S_k \end{pmatrix} \right\rangle \\
&\quad + \left\langle \begin{pmatrix} L - L_k \\ S - S_k \end{pmatrix}, \nabla^2 f \begin{pmatrix} L - L_k \\ S - S_k \end{pmatrix} \right\rangle \\
&= f(X_k) + \left\langle \begin{pmatrix} r_k \\ r_k \end{pmatrix}, \begin{pmatrix} L - L_k \\ S - S_k \end{pmatrix} \right\rangle \\
&\quad + \left\langle \begin{pmatrix} L - L_k \\ S - S_k \end{pmatrix}, \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} L - L_k \\ S - S_k \end{pmatrix} \right\rangle \\
&= f(X_k) + \left\langle \begin{pmatrix} r_k \\ r_k \end{pmatrix}, \begin{pmatrix} L - L_k \\ S - S_k \end{pmatrix} \right\rangle \\
&\quad + \left\langle \begin{pmatrix} \mathbf{L} - L_k \\ \mathbf{S} - S_k \end{pmatrix}, \begin{pmatrix} L - L_k + \mathbf{S} - S_k \\ \mathbf{L} - L_k + S - S_k \end{pmatrix} \right\rangle
\end{aligned}$$

The coupling of the second order terms, shown in bold, prevents direct 1-step minimization of f , subject to the nuclear and 1-norm constraints. The FISTA (Beck & Teboulle, 2009) and spectral gradient methods (SPG) (Wright et al., 2009b) replace the Hessian $\begin{pmatrix} I & I \\ I & I \end{pmatrix}$ with the upper bound $2 \begin{pmatrix} I & 0 \\ 0 & I \end{pmatrix}$, which solves the coupling issue, but potentially lose too much second order information. After comparing FISTA and SPG, we use the SPG method for solving (flip-SPCP_{sum}). However, for (flip-SPCP_{max}) (and for (Lag-SPCP), which has no constraints but rather non-smooth terms, which can be treated like constraints using proximity operators), the constraints are uncoupled and we can take a “middle road” approach, replacing

$$\left\langle \begin{pmatrix} \mathbf{L} - L_k \\ \mathbf{S} - S_k \end{pmatrix}, \begin{pmatrix} L - L_k + \mathbf{S} - S_k \\ \mathbf{L} - L_k + S - S_k \end{pmatrix} \right\rangle$$

with

$$\left\langle \begin{pmatrix} L - L_k \\ S - S_k \end{pmatrix}, \begin{pmatrix} L - L_k + \mathbf{S}_k - \mathbf{S}_{k-1} \\ \mathbf{L}_{k+1} - \mathbf{L}_k + S - S_k \end{pmatrix} \right\rangle.$$

The first term is decoupled, allowing us to update L_k , and then this is plugged into the second term in a Gauss-Seidel fashion. In practice, we also scale this second-order term with a number slightly greater than 1 but less than 2 (e.g., 1.25) which leads to more robust behavior. We expect this “quasi-Newton” trick to do well when $S_{k+1} - S_k$ is similar to $S_k - S_{k-1}$.

7 NUMERICAL RESULTS

The numerical experiments are done with the algorithms suggested in this paper as well as code from PSPG (Aybat et al., 2013), NSA (Aybat & Iyengar, 2014), and ASALM (Tao & Yuan, 2011)². We modi-

²PSPG, NSA and ASALM available from the experiment package at <http://www2.ie.psu.edu/aybat/codes.html>

fied the other software as needed for testing purposes. PSPG, NSA and ASALM all solve (SPCP_{sum}), but ASALM has another variant which solves (Lag-SPCP) so we test this as well. All three programs also use versions of PROPACK from Becker & Candès (2008); Larsen (1998) to compute partial SVDs. Since the cost of a single iteration may vary among the solvers, we measure error as a function of time, not iterations. When a reference solution (L^*, S^*) is available, we measure the (relative) error of a trial solution (L, S) as $\|L - L^*\|_F / \|L^*\|_F + \|S - S^*\|_F / \|S^*\|_F$. The benchmark is designed so the time required to calculate this error at each iteration does not factor into the reported times. Since picking stopping conditions is solver dependent, we show plots of error vs time, rather than list tables. All tests are done in Matlab and the dominant computational time was due to matrix multiplications for all algorithms; all code was run in the same quad-core 1.6 GHz i7 computer.

For our implementations of the (flip-SPCP_{max}), (flip-SPCP_{sum}) and (Lag-SPCP), we use a randomized SVD (Halko et al., 2011). Since the number of singular values needed is not known in advance, the partial SVD may be called several times (the same is true for PSPG, NSA and ASALM). Our code limits the number of singular values on the first two iterations in order to speed up calculation without affecting convergence. Unfortunately, the delicate projection involved in (flip-SPCP_{sum}) makes incorporating a partial SVD to this setting more challenging, so we use Matlab’s dense SVD routine.

7.1 Synthetic test with exponential noise

We first provide a test with generated data. The observations $Y \in \mathbb{R}^{m \times n}$ with $m = 400$ and $n = 500$ were created by first sampling a rank 20 matrix Y_0 with random singular vectors (i.e., from the Haar measure) and singular values drawn from a uniform distribution with mean 0.1, and then adding exponential random noise (with mean equal to one tenth the median absolute value of the entries of Y_0). This exponential noise, which has a longer tail than Gaussian noise, is expected to be captured partly by the S term and partly by the $\|L + S - Y\|_F$ term.

Given Y , the reference solution (L^*, S^*) was generated by solving (Lag-SPCP) to very high accuracy; the values $\lambda_L = 0.25$ and $\lambda_S = 10^{-2}$ were picked by hand tuning (λ_L, λ_S) to find a value such that both L^* and S^* are non-zero. The advantage to solving (Lag-SPCP) is that knowledge of $(L^*, S^*, \lambda_L, \lambda_S)$ allows us to generate the parameters for all the other variants, and hence we can test different problem formulations.

With these parameters, L^* was rank 17 with nuclear

norm 6.754, S^* had 54 non-zero entries (most of them positive) with ℓ_1 norm 0.045, the normalized residual was $\|L^* + S^* - Y\|_F / \|Y\|_F = 0.385$, and $\varepsilon = 1.1086$, $\lambda_{\text{sum}} = 0.04$, $\lambda_{\text{max}} = 150.0593$, $\tau_{\text{sum}} = 6.7558$ and $\tau_{\text{max}} = 6.7540$.

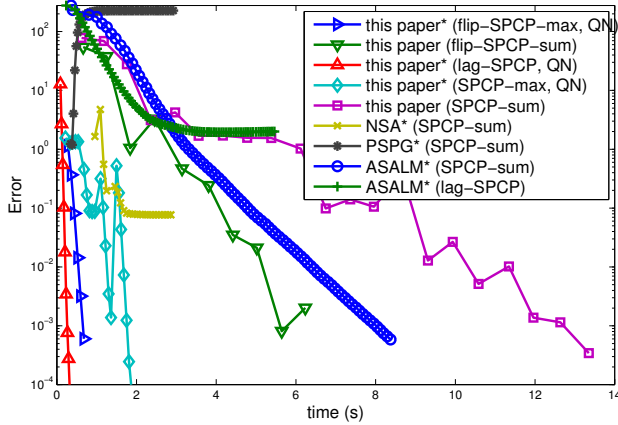


Figure 1: The exponential noise test. The asterisk in the legend means the method uses a fast SVD.

Results are shown in Fig. 1. Our methods for (**flip-SPCP_{max}**) and (**Lag-SPCP**) are extremely fast, because the simple nature of these formulations allows the quasi-Newton acceleration scheme of Section 6. In turn, since our method for solving (**SPCP_{max}**) uses the variational framework of Section 4 to solve a sequence of (**flip-SPCP_{max}**) problems, it is also competitive (shown in cyan in Figure 1). The jumps are due to re-starting the sub-problem solver with a new value of τ , generated according to (4.7).

Our proximal gradient method for (**flip-SPCP_{sum}**), which makes use of the projection in Lemma 5.2, converges more slowly, since it is not easy to accelerate with the quasi-Newton scheme due to variable coupling, and it does not make use of fast SVDs. Our solver for (**SPCP_{sum}**), which depends on a sequence of problems (**flip-SPCP_{sum}**), converges slowly.

The ASALM performs reasonably well, which was unexpected since it was shown to be worse than NSA and PSPG in Aybat et al. (2013); Aybat & Iyengar (2014). The PSPG solver converges to the wrong answer, most likely due to a bad choice of the smoothing parameter μ ; we tried choosing several different values other than the default but did not see improvement for this test (for other tests, not shown, tweaking μ helped significantly). The NSA solver reaches moderate error quickly but stalls before finding a highly accurate solution.

7.2 Synthetic test from Aybat & Iyengar (2014)

We show some tests from the test setup of Aybat & Iyengar (2014) in the $m = n = 1500$ case. The default setting of $\lambda_{\text{sum}} = 1/\sqrt{\max(m, n)}$ was used, and then the NSA solver was run to high accuracy to obtain a reference solution (L^*, S^*) . From the knowledge of $(L^*, S^*, \lambda_{\text{sum}})$, one can generate $\lambda_{\text{max}}, \tau_{\text{sum}}, \tau_{\text{max}}, \varepsilon$, but not λ_S and λ_L , and hence we did not test the solvers for (**Lag-SPCP**) in this experiment. The data was generated as $Y = L_0 + S_0 + Z_0$, where L_0 was sampled by multiplication of $m \times r$ and $r \times n$ normal Gaussian matrices, S_0 had p randomly chosen entries uniformly distributed within $[-100, 100]$, and Z_0 was white noise chosen to give a SNR of 45 dB. We show three tests that vary the rank from $\{0.05, 0.1\} \cdot \min(m, n)$ and the sparsity ranging from $p = \{0.05, 0.1\} \cdot mn$. Unlike Aybat & Iyengar (2014), who report error in terms of a true noiseless signal (L_0, S_0) , we report the optimization error relative to (L^*, S^*) .

For the first test (with $r = 75$ and $p = 0.05 \times mn$), L^* had rank 786 and nuclear norm 111363.9; S^* had 75.49% of its elements nonzero and ℓ_1 norm 5720399.4, and $\|L^* + S^* - Y^*\|_F / \|Y^*\|_F = 1.5 \cdot 10^{-4}$. The other parameters were $\varepsilon = 3.5068$, $\lambda_{\text{sum}} = 0.0258$, $\lambda_{\text{max}} = 0.0195$, $\tau_{\text{sum}} = 2.5906 \cdot 10^5$ and $\tau_{\text{max}} = 1.1136 \cdot 10^5$. An interesting feature of this test is that while L_0 is low-rank, L^* is nearly low-rank but with a small tail of significant singular values until number 786. We expect methods to converge quickly to low-accuracy where only a low-rank approximation is needed, and then slow down as they try to find a larger rank highly-accurate solution.

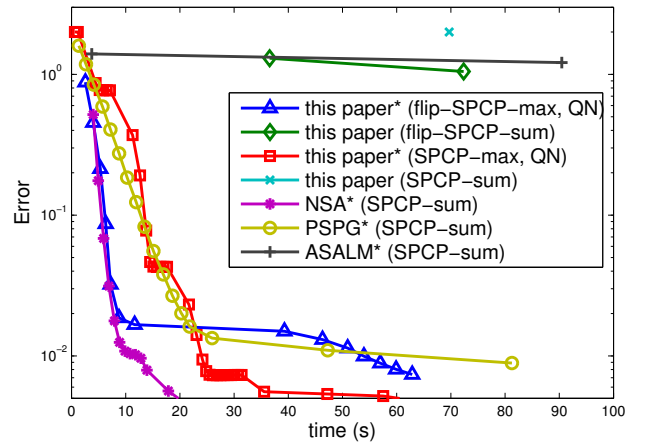


Figure 2: The 1500×1500 synthetic noise test.

The results are shown in Fig. 2. Errors barely dip below 0.01 (for comparison, an error of 2 is achieved

by setting $L = S = 0$). The NSA and PSPG solvers do quite well. In contrast to the previous test, ASALM does poorly. Our methods for ($\text{flip-SPCP}_{\text{sum}}$), and hence (SPCP_{sum}), are not competitive, since they use dense SVDs. We imposed a time-limit of about one minute, so these methods only manage a single iteration or two. Our quasi-Newton method for ($\text{flip-SPCP}_{\text{max}}$) does well initially, then takes a long time due to a long partial SVD computation. Interestingly, (SPCP_{max}) does better than pure ($\text{flip-SPCP}_{\text{max}}$). One possible explanation is that it chooses a fortuitous sequence of τ values, for which the corresponding ($\text{flip-SPCP}_{\text{max}}$) subproblems become increasingly hard, and therefore benefit from the warm-start of the solution of the easier previous problem. This is consistent with empirical observations regarding continuation techniques, see e.g., (van den Berg & Friedlander, 2008; Wright et al., 2009b).

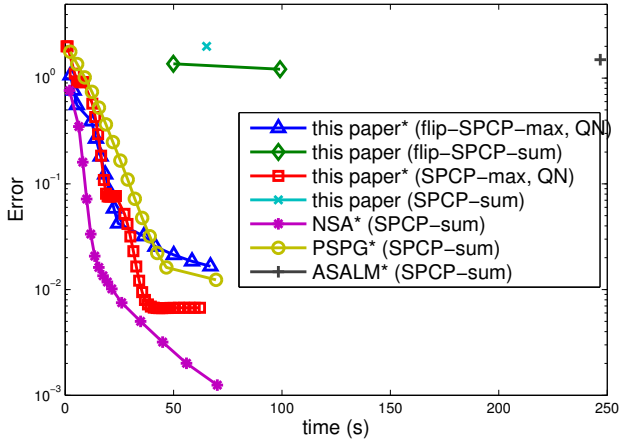


Figure 3: Second 1500×1500 synthetic noise test.

Figure 3 is the same test but with $r = 150$ and $p = 0.1 \cdot mn$, and the conclusions are largely similar.

7.3 Cloud removal

Figure 4 shows 15 images of size 300×300 from the MODIS satellite,³ after some transformations to turn images from different spectral bands into one grayscale images. Each image is a photo of the same rural location but at different points in time over the course of a few months. The background changes slowly and the variability is due to changes in vegetation, snow cover, and different reflectance. There are also outlying sources of error, mainly due to clouds (e.g., major clouds in frames 5 and 7, smaller clouds in frames 9, 11 and 12), as well as artifacts of the CCD camera on

³Publicly available at <http://ladsweb.nascom.nasa.gov/>

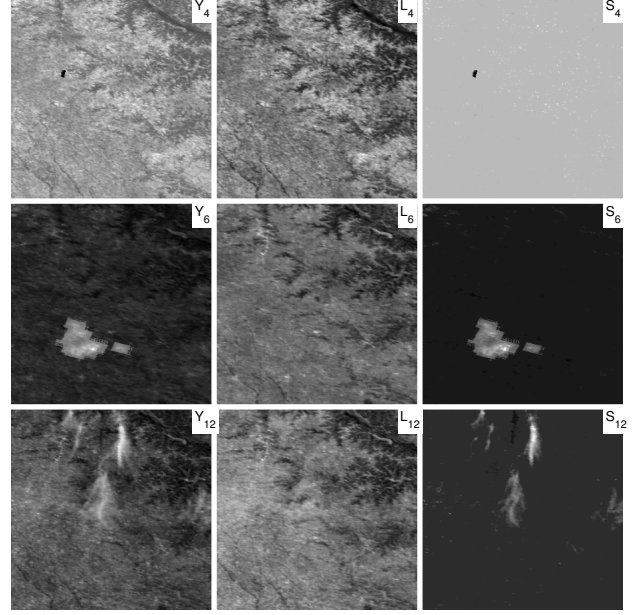


Figure 5: Showing frames 4, 5 and 12. Leftmost column is original data, middle column is low-rank term of the solution, and right column is sparse term of the solution. Data have been processed slightly to enhance contrast for viewing.

the satellite (frame 4 and 6) and issues stitching together photos of the same scene (the lines in frames 8 and 10).

There are hundreds of applications for clean satellite imagery, so removing the outlying error is of great practical importance. Because of slow changing background and sparse errors, we can model the problem using the robust PCA approach. We use the ($\text{flip-SPCP}_{\text{max}}$) version due to its speed, and pick parameters ($\lambda_{\text{max}}, \tau_{\text{max}}$) by using a Nelder-Mead simplex search. For an error metric to use in the parameter tuning, we remove frame 1 from the data set (call it y_1) and set Y to be frames 2–15. From this training data Y , the algorithm generates L and S . Since L is a $300^2 \times 14$ matrix, it has far from full column span. Thus our error is the distance of y_1 from the span of L , i.e., $\|y_1 - \mathcal{P}_{\text{span}(L)}(y_1)\|_2$.

Our method takes about 11 iterations and 5 seconds, and uses a dense SVD instead of the randomized method due to the high aspect ratio of the matrix. Some results of the obtained (L, S) outputs are in Fig. 5, where one can see that some of the anomalies in the original data frames Y are picked up by the S term and removed from the L term. Frame 4 has what appears to be a camera pixel error; frame 6 has another artificial error (that is, caused by the camera and not the scene); and frame 12 has cloud cover.

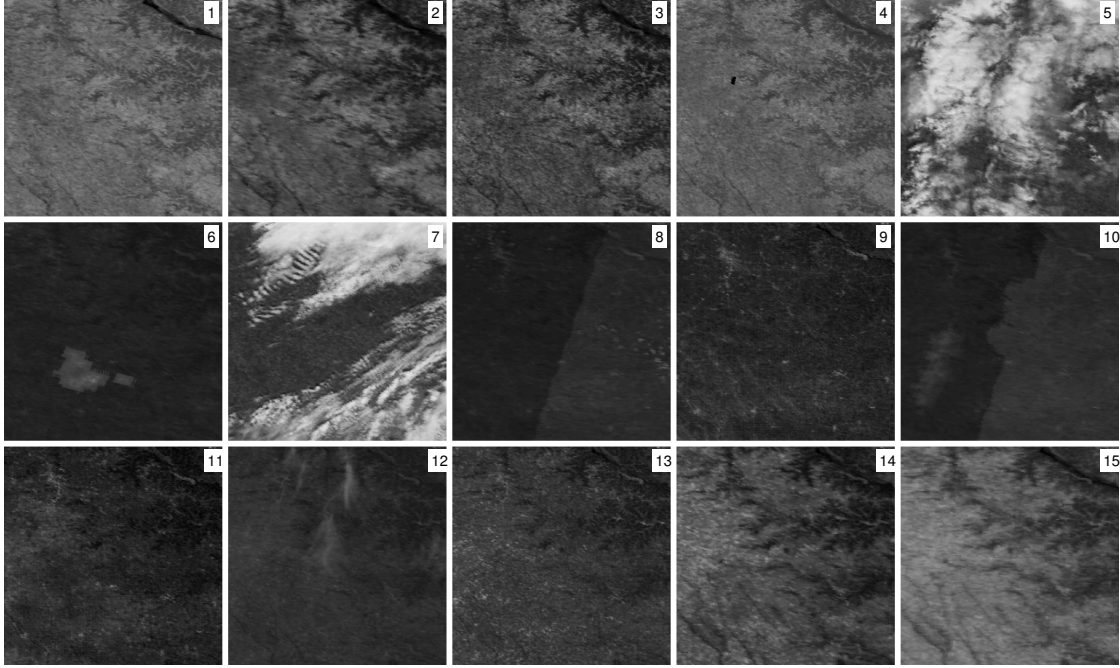


Figure 4: Satellite photos of the same location on different days

8 CONCLUSIONS

In this paper, we reviewed several formulations and algorithms for the RPCA problem. We introduced a new denoising formulation (SPCP_{\max}) to the ones previously considered, and discussed modeling and algorithmic advantages of denoising formulations (SPCP_{\max}) and (SPCP_{sum}) compared to flipped versions (flip-SPCP_{\max}) and ($\text{flip-SPCP}_{\text{sum}}$). In particular, we showed that these formulations can be linked using a variational framework, which can be exploited to solve denoising formulations using a sequence of flipped problems. For (flip-SPCP_{\max}), we proposed a quasi-Newton acceleration that is competitive with state of the art, and used this innovation to design a fast method for (SPCP_{\max}) through the variational framework. The new methods were compared against prior art on synthetic examples, and applied to a real world cloud removal application using publicly available MODIS satellite data.

References

- Aravkin, A. Y., Burke, J., and Friedlander, M. P. Variational properties of value functions. *SIAM J. Optimization*, 23(3):1689–1717, 2013a.
- Aravkin, A. Y., Kumar, R., Mansour, H., Recht, B., and Herrmann, F. J. A robust SVD-free approach to matrix completion, with applications to interpolation of large scale data. 2013b. URL <http://arxiv.org/abs/1302.4886>.
- Aybat, N., Goldfarb, D., and Ma, S. Efficient algorithms for robust and stable principal component pursuit. *Computational Optimization and Applications*, (accepted), 2013.
- Aybat, N. S. and Iyengar, G. An alternating direction method with increasing penalty for stable principal component pursuit. *Computational Optimization and Applications*, (submitted), 2014. <http://arxiv.org/abs/1309.6553>.
- Baldassarre, L., Cevher, V., McCoy, M., Tran Dinh, Q., and Asaei, A. Convexity in source separation: Models, geometry, and algorithms. Technical report, 2013. <http://arxiv.org/abs/1311.0258>.
- Beck, A. and Teboulle, M. A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. *SIAM J. Imaging Sciences*, 2(1):183–202, January 2009.
- Becker, S. and Candès, E. Singular value thresholding toolbox, 2008. Available from <http://svt.stanford.edu/>.
- Brucker, P. An $O(n)$ algorithm for quadratic knapsack problems. *Operations Res. Lett.*, 3(3):163 – 166, 1984. doi: 10.1016/0167-6377(84)90010-5.
- Candès, E. J., Li, X., Ma, Y., and Wright, J. Robust principal component analysis? *J. Assoc. Comput. Mach.*, 58(3):1–37, May 2011.
- Chandrasekaran, V., Sanghavi, S., Parrilo, P. A., and Willsky, A. S. Sparse and low-rank matrix decompositions. In *SYSID 2009*, Saint-Malo, France, July 2009.
- Chandrasekaran, V., Parrilo, P. A., and Willsky, A. S. Latent variable graphical model selection via convex optimization. *Ann. Stat.*, 40(4):1935–2357, 2012.
- Chen, Caihua, He, Bingsheng, Ye, Yinyu, and Yuan, Xiaoming. The direct extension of admm for multi-block convex minimization problems is not necessarily convergent. *Optimization Online*, 2013.

- Combettes, P. L. and Pesquet, J.-C. A Douglas–Rachford Splitting Approach to Nonsmooth Convex Variational Signal Recovery. *IEEE J. Sel. Topics Sig. Processing*, 1(4):564–574, December 2007.
- Duchi, J., Shalev-Shwartz, S., Singer, Y., and Chandra, T. Efficient projections onto the ℓ_1 -ball for learning in high dimensions. In *Intl. Conf. Machine Learning (ICML)*, pp. 272–279, New York, July 2008. ACM Press.
- Ganesh, A., Lin, Z., Wright, J., Wu, L., Chen, M., and Ma, Y. Fast algorithms for recovering a corrupted low-rank matrix. In *Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pp. 213–215, Aruba, Dec. 2009.
- Halko, N., Martinsson, P.-G., and Tropp, J. A. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.
- Huber, P. J. *Robust Statistics*. John Wiley and Sons, 2 edition, 2004.
- Kyrillidis, Anastasios and Cevher, Volkan. Matrix recipes for hard thresholding methods. *Journal of Mathematical Imaging and Vision*, 48(2):235–265, 2014.
- Larsen, R. M. Lanczos bidiagonalization with partial reorthogonalization. Tech. Report. DAIMI PB-357, Department of Computer Science, Aarhus University, September 1998.
- Lee, J., Recht, B., Salakhutdinov, R., Srebro, N., and Tropp, J.A. Practical large-scale optimization for max-norm regularization. In *Neural Information Processing Systems (NIPS)*, Vancouver, 2010.
- Lin, Z., Chen, M., and Ma, Y. The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices. *arXiv preprint arXiv:1009.5055*, 2010.
- Mansour, H. and Vetro, A. Video background subtraction using semi-supervised robust matrix completion. In *To appear in IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2014.
- Peng, Y., Ganesh, A., Wright, J., Xu, W., and Ma, Y. RASL: Robust alignment by sparse and low-rank decomposition for linearly correlated images. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 34(11):2233–2246, 2012.
- Recht, Benjamin and Ré, Christopher. Parallel stochastic gradient algorithms for large-scale matrix completion. *Math. Prog. Comput.*, pp. 1–26, 2011.
- Rockafellar, R. T. *Convex analysis*. Princeton Mathematical Series, No. 28. Princeton University Press, Princeton, N.J., 1970.
- Shen, Y., Wen, Z., and Zhang, Y. Augmented Lagrangian alternating direction method for matrix separation based on low-rank factorization. *Optimization Methods and Software*, 29(2):239–263, March 2014.
- Tao, M. and Yuan, X. Recovering low-rank and sparse components of matrices from incomplete and noisy observations. *SIAM J. Optimization*, 21:57–81, 2011.
- van den Berg, E. and Friedlander, M. P. Probing the Pareto frontier for basis pursuit solutions. *SIAM J. Sci. Computing*, 31(2):890–912, 2008. software: <http://www.cs.ubc.ca/~mpf/spgl1/>.
- van den Berg, E. and Friedlander, M. P. Sparse optimization with least-squares constraints. *SIAM J. Optimization*, 21(4):1201–1229, 2011.
- Wright, J., Ganesh, A., Rao, S., and Ma, Y. Robust principal component analysis: Exact recovery of corrupted low-rank matrices by convex optimization. In *Neural Information Processing Systems (NIPS)*, 2009a.
- Wright, S. J., Nowak, R. D., and Figueiredo, M. A. T. Sparse Reconstruction by Separable Approximation. *IEEE Trans. Sig. Processing*, 57(7):2479–2493, July 2009b.
- Zhang, Z., Liang, X., Ganesh, A., and Ma, Y. TILT: Transform invariant low-rank textures. In Kimmel, R., Klette, R., and Sugimoto, A. (eds.), *Computer Vision – ACCV 2010*, volume 6494 of *Lecture Notes in Computer Science*, pp. 314–328. Springer, 2011.

Markov Network Structure Learning via Ensemble-of-Forests Models

Eirini Arvaniti¹ and Manfred Claassen^{1†}

¹Institute of Molecular Systems Biology, ETH Zurich, Zurich, Switzerland.

[†]Electronic correspondence: claassen@imsb.biol.ethz.ch

Abstract

Real world systems typically feature a variety of different dependency types and topologies that complicate model selection for probabilistic graphical models. We introduce the *ensemble-of-forests* model, a generalization of the *ensemble-of-trees* model of Meilä and Jaakkola (2006). Our model enables structure learning of Markov random fields (MRF) with multiple connected components and arbitrary potentials. We present two approximate inference techniques for this model and demonstrate their performance on synthetic data. Our results suggest that the ensemble-of-forests approach can accurately recover sparse, possibly disconnected MRF topologies, even in presence of non-Gaussian dependencies and/or low sample size. We applied the ensemble-of-forests model to learn the structure of perturbed signaling networks of immune cells and found that these frequently exhibit non-Gaussian dependencies with disconnected MRF topologies. In summary, we expect that the ensemble-of-forests model will enable MRF structure learning in other high dimensional real world settings that are governed by non-trivial dependencies.

1 INTRODUCTION

This work presents the ensemble-of-forests model for approximate structure learning in Markov random fields (MRF). As opposed to most existing MRF structure learners that either work with specific types of potentials (e.g. discrete, Gaussian) or assume connected MRF topology (Lin et al., 2009), our approach is applicable for MRFs with arbitrary potentials and topology, including disconnected topologies, and is therefore suited to accommodate a wide range of real world settings.

Markov random fields (MRF) are undirected probabilistic graphical models specifying conditional independence relations among a set of random variables. Learning MRFs involves parameter inference and model selection, i.e. learning the underlying graph structure. For general MRFs, exact parameter inference is difficult due to the necessity to evaluate the intractable partition sum and therefore is addressed by approximate inference approaches. Structure learning is an even more difficult task. The naive method of enumerating all possible topologies is prohibitively expensive and, thus, alternative approaches have been proposed based on independence tests or approximate score-based methods Koller and Friedman (2009).

Currently, the prevalent approach to model continuous random variables is to assume Gaussianity. Under this hypothesis, the Gaussian Markov random field (GMRF) structure can be directly read from the inverse covariance matrix (Koller and Friedman, 2009): zero entries exactly correspond to conditional independence statements of the Markov random field. Sparse inverse covariance selection constitutes a convex relaxation of the structure learning task for GMRFs that can be solved efficiently (Banerjee et al., 2006; Friedman et al., 2008).

Random variables of real world systems typically exhibit unusual dependency types (Trivedi and Zimmer, 2005; Berkes et al., 2008) that are not appropriately captured by the Gaussian potentials of GMRFs. *Copula* potentials constitute a more general and expressive alternative to deal with non-Gaussian dependency types. Copulas are multivariate distributions that encode the dependencies among random variables. Copula models are very flexible, as they enable researchers to independently specify the marginal distributions of random variables and their dependency structure. Liu et al. (2009) define MRFs with semi-parametric Gaussian copula potentials. Approximate structure learning in this model is tractable because the dependency type is Gaussian and, thus, parameter inference is easy and model selection can also be efficiently approximated by resorting to sparse inverse covariance estimation. However, in MRFs with general copula potentials, even

parameter estimation is difficult because of the intractable partition sum. This situation entails that structure learning is also difficult.

The intractability of exact inference for MRFs with general copula potentials has motivated alternative approaches based on approximate inference. Meilä and Jaakkola (2006) introduced the *ensemble-of-trees* (ET) model that enables approximate inference for both parameter estimation and structure learning of general MRFs. A Markov network is represented as a mixture model whose components are tree-structured distributions defined over all possible spanning trees of the underlying graph. Despite the super-exponential number of such trees, the model remains tractable by defining conveniently decomposable priors over the structure and parameters of tree-distributions. Recently, Kirshner (2008) presented a tree-averaged density model based on tree structured MRFs with copula potentials. The tasks of parameter estimation and structure learning are jointly expressed as a single (non-convex) objective, which is optimized via Expectation-Maximization. Lin et al. (2009) utilize the ET model for structure learning of GMRFs and empirically demonstrate superior performance compared to sparse inverse covariance selection for limited sample size. Above considerations render copula MRFs as attractive models because they are more general than GMRFs and efficient learning approaches exist for them.

Real world systems with many random variables are frequently best represented by MRFs that decompose into several connected components. In biology, for instance, a specific stimulus might activate competing, independent signaling pathways each including its own MRF component (Johnstone et al., 2008). However, the ET structure learning approach is not able to recover disconnected topologies since it is averaging over ensembles of spanning trees. It is desirable to generalize the ET approach in order to overcome this limitation and, thereby, still benefit from the expressiveness of copula MRFs in these real world settings.

The main contribution of this work is the generalization of the ET model to the *ensemble-of-forests* (EF) model that explicitly accounts for graph topologies with multiple connected components. In the proposed model, a Markov network is represented as a mixture of forests, i.e. collections of tree-structured MRFs. An implementation of the exact model is intractable, as the averaging over all possible forests results in a hard combinatorial problem. Instead, we present approximate formulations of the structure learning task. The rest of this paper is organized as follows. In Sections 2 – 3 we formally introduce the methods that we build upon. Then, in Sections 4 – 6 we describe the *ensemble-of-forests* model and present benchmark results on synthetic datasets. In Sections 7 – 8 we apply our method to plant microarray and immune cell perturbation data. Finally, Section 9 concludes with a short discussion.

2 COPULA MODELS

This section reviews the application of copulas to describe general multivariate distributions and/or potentials in MRFs. Copulas are multivariate continuous distributions defined on the unit hypercube, $C : [0, 1]^d \rightarrow [0, 1]$, with uniform univariate marginals. Let X_1, \dots, X_d be real random variables with joint cumulative distribution function (cdf) $F(\mathbf{x})$ and marginally distributed as $F_1(x_1), \dots, F_d(x_d)$ respectively. Then, the random variables $U_1 = F_1(x_1), \dots, U_d = F_d(x_d)$ are uniformly distributed on $[0, 1]$. This property forms the basis for Sklar’s theorem, according to which any joint distribution $F(x_1, \dots, x_d)$ with continuous marginals can be uniquely expressed as

$$F(x_1, \dots, x_d) = C(F_1(x_1), \dots, F_d(x_d)). \quad (1)$$

The converse is also true: arbitrary univariate marginals $\{F_i\}$ can be combined using a copula function C to uniquely construct a valid joint distribution with marginals $\{F_i\}$. The copula function C exclusively encodes the dependencies among random variables.

Furthermore, copula density functions $c(\mathbf{u}) = \frac{\partial^d C(\mathbf{u})}{\partial u_1 \dots \partial u_d}$ can be expressed in terms of probability density functions as

$$c(u_1, \dots, u_d) = \frac{f(x_1, \dots, x_d)}{\prod_{i=1}^d f_i(x_i)}. \quad (2)$$

A large number of copula functions have been proposed in the literature (Nelsen, 1999), especially for the bivariate case. Commonly used examples are the Clayton, Gumbel, Frank, Gaussian and Student’s t parametric copula families. In Figure 1, we present contour plots of six distributions with standard Gaussian marginals but different types of dependencies between the marginals. In each case, the dependency structure is specified via a different copula function.

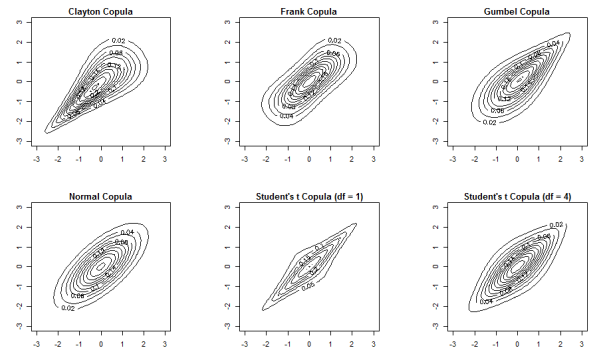


Figure 1: Contour plots of six joint distributions defined using standard Gaussian marginals and different dependency structures specified by different copulas.

Bivariate copulas are typically used to model strong extreme-value dependencies in financial data (Embrechts et al., 2003; Trivedi and Zimmer, 2005). Recently, the probabilistic graphical model framework has been successfully employed for the construction of copula-based high-dimensional models. A review on this topic can be found in (Elidan, 2013).

3 ENSEMBLE-OF-TREES MODELS

Here we introduce the ensemble-of-trees (ET) method for approximate parameter inference and structure learning of MRFs. This method forms the basis for the ensemble-of-forests method, the main conceptual contribution of this paper. From here on, we adopt the following notation: we consider a Markov network encoded by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where \mathcal{V} is the set of nodes corresponding to random variables $\mathcal{X} = \{X_1, \dots, X_d\}$ and \mathcal{E} is the set of edges.

The ensemble-of-trees model of Meilä and Jaakkola (2006) is an approximate inference approach to carry out structure learning for MRFs with “inconvenient” potentials. It constitutes a mixture model over all possible spanning trees of the complete graph over the nodeset \mathcal{V} . A prior distribution over spanning tree structures T is defined as

$$p_\beta(T) = \frac{1}{Z_\beta} \prod_{e_{uv} \in T} \beta_{uv} \quad (3)$$

where each parameter $\beta_{uv} = \beta_{vu} \geq 0$, for all $u \neq v$, $u, v \in \mathcal{V}$ can be interpreted as a weight for edge e_{uv} , directly proportional to the probability of appearance of that edge.

$Z_\beta = \sum_T \prod_{e_{uv} \in T} \beta_{uv}$ is a normalizing constant, ensuring that the prior constitutes a valid probability distribution. It turns out that Z_β can be efficiently computed. Defining the matrix $\mathbf{Q}(\beta)$ as the first $d-1$ rows and columns of the Laplacian matrix

$$L_{uv} = \begin{cases} -\beta_{uv} & \text{if } u \neq v, \\ \sum_k \beta_{uk} & \text{if } u = v \end{cases} \quad (4)$$

Meilä and Jaakkola (2006) generalize Kirchhoff’s Matrix-Tree theorem for binary weights and show that

$$Z_\beta = \sum_T \prod_{e_{uv} \in T} \beta_{uv} = |\mathbf{Q}(\beta)|. \quad (5)$$

This result makes the averaging over all possible (d^{d-2}) spanning tree structures computationally tractable.

Assuming a prior tree structure T , the conditional distribution of a data sample \mathbf{x} can be expressed as

$$p(\mathbf{x}|T, \theta) = \prod_{v \in \mathcal{V}} \theta_v(x_v) \prod_{e_{uv} \in T} \frac{\theta_{uv}(x_u, x_v)}{\theta_u(x_u)\theta_v(x_v)} \quad (6)$$

where the parameter vector θ consists of univariate $\theta_v(x_v)$ and bivariate $\theta_{uv}(x_u, x_v)$ marginal densities defined, respectively, over the nodes and the edges of the tree (Meilä and Jaakkola, 2006). These distributions are assumed invariant for all tree structures.

Finally, after introducing the notation $w_{uv}(\mathbf{x}) = \frac{\theta_{uv}(x_u, x_v)}{\theta_u(x_u)\theta_v(x_v)}$, $w_0(\mathbf{x}) = \prod_{v \in \mathcal{V}} \theta_v(x_v)$ and applying twice the generalized Matrix-Tree theorem we have

$$\begin{aligned} p_\beta(\mathbf{x}) &= \sum_T p_\beta(T) p(\mathbf{x}|T, \theta) \\ &= \frac{w_0(\mathbf{x})}{Z_\beta} \sum_T \prod_{e_{uv} \in T} \beta_{uv} w_{uv}(\mathbf{x}) \\ &= w_0(\mathbf{x}) \frac{|\mathbf{Q}(\beta \otimes \mathbf{w}(\mathbf{x}))|}{|\mathbf{Q}(\beta)|} \quad (7) \end{aligned}$$

where the symbol \otimes denotes element-wise multiplication.

The structure learning task in the ET model can be approximated by an empirical estimation of β , as in (Lin et al., 2009), where β is used to approximate the MRF adjacency matrix: non-zero entries β_{uv} correspond to edges in the graph. In our model, we adopt this interpretation of β .

3.1 ET MODELS WITH DISCONNECTED SUPPORT GRAPH

A mixture model over spanning trees is based on the implicit assumption that the *support graph* of the model is connected. The support graph is a graph that contains exactly the edges corresponding to positive entries in β . The case of disconnected support graphs is considered by Meilä and Jaakkola (2006) only for *a priori* defined connected components. That is, certain patterns of zero entries in the parameter set β predefine a partitioning of nodes into different connected components and these assignments to components cannot be changed e.g. during the course of a structure learning procedure. In this case, each connected component can be treated independently from all others. Assuming k connected components that partition \mathcal{V} into $\{V^1, \dots, V^k\}$ and introducing the notation

$$\beta_{V^i} = \{\beta_{uv}, u \neq v, u, v \in V^i\}$$

equation (7) is generalized as

$$p_\beta(\mathbf{x}) = w_0(\mathbf{x}) \frac{\prod_{i=1}^k |\mathbf{Q}(\beta_{V^i} \otimes \mathbf{w}_{V^i}(\mathbf{x}))|}{\prod_{i=1}^k |\mathbf{Q}(\beta_{V^i})|} \quad (8)$$

4 ENSEMBLE-OF-FORESTS MODELS

Here we introduce the main contribution of our work, that is the *ensemble-of-forests* (EF) model. This model constitutes an approximate inference approach for structure

learning of MRFs with multiple connected components that are not known *a priori*. We assume a nodeset \mathcal{V} of size d and a partition thereof $\mathbf{V} = \{V^1, \dots, V^k\}$. Then, a *maximal forest* or *forest* of size k is a collection of spanning trees $\{T^i\}_{i=1, \dots, k}$, one for each V^i . Extending the ensemble-of-trees model, we introduce a mixture model over all possible forests up to a certain size, i.e. allowing for disconnected structures with a maximal number of k connected components. The limiting cases are $k = 1$, corresponding to the ET model, and $k = d$, corresponding to a model that allows for any possible arrangement of connected components.

The prior probability of a collection of spanning trees $\mathcal{F} := \{T^1, \dots, T^k\}$ is defined as

$$p_{\beta}(\mathcal{F}) = \frac{1}{Z_{\beta}} \prod_{T^i \in \mathcal{F}} \prod_{e_{uv} \in T^i} \beta_{uv} \quad (9)$$

where $\beta_{uv} = \beta_{vu} \geq 0$, for all $u \neq v$, $u, v \in \mathcal{V}$. Now, in order to normalize over all possible forests that consist of at most k connected components, the partition function is computed via

$$\begin{aligned} Z_{\beta} &= \sum_{\mathbf{V} \in \text{part}(\mathcal{V})} \sum_{\mathcal{F} \in f(\mathbf{V})} \prod_{T^i \in \mathcal{F}} \prod_{e_{uv} \in T^i} \beta_{uv} \\ &= \sum_{\mathbf{V} \in \text{part}(\mathcal{V})} \prod_{V^i \in \mathbf{V}} |\mathbf{Q}(\beta_{V^i})| \quad (10) \end{aligned}$$

where the outer summation $\sum_{\mathbf{V} \in \text{part}(\mathcal{V})}$ is performed over all possible partitions of \mathcal{V} into k subsets and the inner summation $\sum_{\mathcal{F} \in f(\mathbf{V})}$ is performed over all maximal forests defined on a specific node partition \mathbf{V} . Partitions where some of the subsets V^i are empty are allowed and correspond to graphs with less than k connected components. For example, the partition $\{\mathcal{V}, \emptyset, \dots, \emptyset\}$ represents a fully connected graph. In order to treat such partitions without changing our notation, we define $\mathbf{Q}(\beta_{\emptyset}) = 1$.

Ignoring the constant term $w_0(\mathbf{x})$, the negative log-likelihood of the model given a dataset $\mathcal{D} = \{x^{(1)}, \dots, x^{(N)}\}$ is written as

$$\begin{aligned} \mathcal{L}(\mathcal{D}; \beta) &= N \log \sum_{\mathbf{V} \in \text{part}(\mathcal{V})} \prod_{V^i \in \mathbf{V}} |\mathbf{Q}(\beta_{V^i})| \\ &\quad - \sum_{j=1}^N \log \sum_{\mathbf{V} \in \text{part}(\mathcal{V})} \prod_{V^i \in \mathbf{V}} |\mathbf{Q}(\beta \mathbf{w}_{V^i}^{(j)})| \quad (11) \end{aligned}$$

where $\beta \mathbf{w}_{V^i}^{(j)}$ is a shorthand for $\beta_{V^i} \otimes \mathbf{w}_{V^i}(\mathbf{x}^{(j)})$.

5 LEARNING IN THE EF MODEL

In this section, we describe two approaches for structure learning of Markov networks based on the EF model,

namely the *EF-cuts* and *EF- λ* methods. Additionally, we describe common features of the two methods, such as the choice of MRF potentials and the optimization algorithm used for minimizing the learning objective.

5.1 SELECTION OF EDGE POTENTIALS

The first step in learning the EF model is concerned with the choice of the edge potentials $w_{uv}(\mathbf{x})$. Here, we consider continuous distributions as edge potentials. Although we do not explicitly consider discrete distributions in the following, we want to emphasize that learning in the EF model easily extends to this class of potentials. In order to keep our model as generic as possible, we have chosen to use copula-based potentials. Note from Equation (2) that the potentials $w_{uv}(\mathbf{x})$ exactly correspond to bivariate copula densities. In our analysis, we have used the bivariate Clayton, Frank, Gumbel, Gaussian and Student's t copula as candidate parametric families. These copulas have one single parameter to be estimated.

In order to fit a single-parameter copula family to data, we follow a two-step procedure. As a first step, the marginal cdf for each random variable is estimated in a non-parametric approach (Kojadinovic and Yan, 2010) and the obtained estimators, known as pseudo-observations, are plugged into the copula function. Subsequently, the dependence parameter is computed by maximizing the pseudo-likelihood

$$\log L(\theta) = \sum_{i=1}^n \log c(\hat{\mathbf{u}}_i; \theta) \quad (12)$$

where $\hat{\mathbf{U}}_i$ is the vector of estimators for the marginals and n is the sample size. The best-fitting copula for each variable pair is selected via cross-validation, where the cross-validation score is based on the pseudo-likelihood of the left-out samples.

5.2 THE EF-cuts HEURISTIC

Graphs with two connected components constitute an important subclass of disconnected networks. Even when restricting ourselves to a maximum of two connected components, it is computationally prohibitive to use the exact ensemble-of-forests model of Equation (11) for sets of random variables of non-trivial size due to the super-exponential number of possible node partitions $\text{part}(\mathcal{V})$. Therefore, we resort to heuristic approaches for choosing partitions that are most likely to allow us to recover the true graph structure. For a given parameter configuration β , we aim to identify a number of high scoring partitions of the nodeset and then average over these partitions only.

Our heuristic is based on the intuition that edges e_{uv} with small β_{uv} are assigned a low prior probability and, therefore, are expected to be most likely not present in the true

MRF. Therefore, we would like to prioritize partitions generated by dropping these low-weight edges. Following that intuition, we derive a scoring system based on systematic enumeration of minimum cuts.

A *cut* of a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a partition of \mathcal{V} into subsets $A, B = \mathcal{V} - A$. The weight of a cut is the sum of the weights of all edges crossing the cut. Starting with the minimum-weight cut, we want to enumerate a ranked set of graph cuts of increasing weight. An efficient algorithm (Vazirani and Yannakakis, 1992) exists for this task. In our case, edge weights correspond to the structural parameters β . Let (A, B) denote a cut and let \mathcal{C} denote the set of M minimum-weight cuts in the graph. Since we are only considering graphs with at most two connected components, a forest \mathcal{F} consists of two spanning trees T_A, T_B . To simplify our notation, we include the case of connected graphs as a special case where $A = \mathcal{V}$ and $B = \emptyset$. This is a special cut of zero weight and is always included in \mathcal{C} . We perform structure learning by minimizing the negative log-likelihood of the model with respect to β . The respective objective is derived from Equation (11) by setting $k = 2$ and only considering partitions that belong to the set \mathcal{C} . The optimization problem can be formulated as

$$\begin{aligned} \min_{\beta} N \log \sum_{(A,B) \in \mathcal{C}} |\mathbf{Q}(\beta_A)| |\mathbf{Q}(\beta_B)| \\ - \sum_{j=1}^N \log \sum_{(A,B) \in \mathcal{C}} |\mathbf{Q}(\beta \mathbf{w}_A^{(j)})| |\mathbf{Q}(\beta \mathbf{w}_B^{(j)})| \\ \text{s.t. } \beta_{uv} \geq 0 \quad u, v \in \mathcal{V}, \quad u \neq v. \end{aligned} \quad (13)$$

Let us denote \mathcal{C}' the set of partitions where nodes u, v belong to the same connected component. The set of partitions where u, v belong to different components has no contribution to the gradient $(\nabla_{\beta} f)_{uv}$. Without loss of generality, we will assume that if nodes u, v belong to the same partition set, then this is set A and the other set is $B = \mathcal{V} - A$. Then the gradient of the objective (13) follows as

$$\begin{aligned} (\nabla_{\beta} f)_{uv} = N \frac{\sum_{(A,B) \in \mathcal{C}'} M_{uv}(\beta_A) |\mathbf{Q}(\beta_A)| |\mathbf{Q}(\beta_B)|}{\sum_{(A,B) \in \mathcal{C}'} |\mathbf{Q}(\beta_A)| |\mathbf{Q}(\beta_B)|} \\ - \sum_{j=1}^N w_{uv}^{(j)} \frac{\sum_{(A,B) \in \mathcal{C}'} M_{uv}(\beta_A) |\mathbf{Q}(\beta \mathbf{w}_A^{(j)})| |\mathbf{Q}(\beta \mathbf{w}_B^{(j)})|}{\sum_{(A,B) \in \mathcal{C}'} |\mathbf{Q}(\beta \mathbf{w}_A^{(j)})| |\mathbf{Q}(\beta \mathbf{w}_B^{(j)})|} \end{aligned} \quad (14)$$

where M is defined as in (Meilă and Jaakkola, 2006)

$$M_{uv} = \begin{cases} Q_{uu}^{-1} + Q_{vv}^{-1} - 2Q_{uv}^{-1} & \text{if } u \neq v, u \neq w, v \neq w, \\ Q_{uu}^{-1} & \text{if } u \neq v, v = w, \\ Q_{vv}^{-1} & \text{if } u \neq v, u = w, \\ 0 & \text{if } u = v. \end{cases} \quad (15)$$

With w we denote the index of the row and column that are removed from the Laplacian matrix of Equation (4) in order to obtain Q .

The min-cut heuristic is a feasible approximation to structure learning of MRFs with disconnected topologies. However, it is practically restricted to graph structures with at most two connected components. Furthermore, the approach does not scale with increasing node or sample size due to the complicated objective and gradient functions. These considerations limit its applicability to real world scenarios.

5.3 THE EF- λ HEURISTIC

In the following, we introduce the EF- λ heuristic that scales well with dimensionality and number of connected components of the underlying MRF. The starting point is again equation (11), but now we drop the summation $\sum_{\mathbf{V} \in \text{part}(\mathcal{V})}$ over possible node partitions. Instead, we only consider a single partition \mathbf{V} . Additionally, we impose an L_1 penalty term on the structural parameters β to encourage sparse solutions. The new optimization task is expressed as

$$\begin{aligned} \min_{\beta} N \sum_{V^i \in \mathbf{V}} \log |\mathbf{Q}(\beta_{V^i})| - \sum_{j=1}^N \sum_{V^i \in \mathbf{V}} \log |\mathbf{Q}(\beta \mathbf{w}_{V^i}^{(j)})| + \lambda \|\beta\|_1 \\ \text{s.t. } \beta_{uv} \geq 0 \quad u, v \in \mathcal{V}, \quad u \neq v. \end{aligned} \quad (16)$$

An iterative optimization procedure is employed to minimize the objective (16). At each iteration step, summation is performed over maximal forests defined for the single node partition \mathbf{V} that is induced by the current iterate β . The number of connected components does not need to be fixed. The penalty term has the critical role of controlling sparsity and, thus, allowing structures with multiple connected components to be considered.

A similar L_1 -regularized approach cannot be employed for the ET model, because the ET objective is not defined for all sparsity patterns in β . Therefore, there is effectively no sparsity induction by an L_1 penalty in ET. Furthermore, for some iterative optimization procedures, numerical instabilities might occur if β is temporarily set to an invalid value.

The gradient of the objective for the EF- λ takes a simple form. Considering the non-negativity of β , the L_1 -norm $\|\beta\|_1$ is equal to $\sum_{u,v \in \mathcal{V}, u \neq v} \beta_{uv}$. Thus, the objective is differentiable at all points. Assuming that β induces a par-

tioning of \mathcal{V} into $\{V^1, \dots, V^k\}$, the gradient of the objective can be expressed as

$$(\nabla_{\beta} f)_{uv} = NM_{uv}(\beta_{V^i}) - \sum_{j=1}^N w_{uv}^{(j)} M_{uv}(\beta \mathbf{w}_{V^i}^{(j)}) + \lambda \quad (17)$$

for $u, v \in V^i$ and is equal to 0 otherwise.

The choice of the regularization parameter λ is an important aspect of the EF- λ approach. We optimize the EF- λ objective using different penalty parameters $\lambda = \exp(-\rho)$, where ρ takes values in the interval $[3, 6]$ with a step of 0.1. The optimal λ is selected so as to minimize the extended Bayesian Information Criterion (eBIC) (Foygel and Drton, 2010) defined as

$$eBIC = 2\mathcal{L} + |E| \log n + 4|E|\gamma \log d \quad (18)$$

where \mathcal{L} is the negative log-likelihood of the model, $|E|$ is the number of non-zero predicted β entries, n is the sample size, d is the number of nodes and γ is an additional penalty term imposed on more complex structures. The classical Bayesian Information Criterion is obtained as a subcase for $\gamma = 0$. We performed simulations with different values of γ in the interval $[0, 1]$ and resulted in using $\gamma = 0.5$.

5.4 OPTIMIZATION OF THE LEARNING OBJECTIVE

The objectives (13) and (16) to fit the EF model are non-convex functions. Therefore, there is no guarantee of convergence to a global optimum and the initial point for optimization has to be carefully chosen. Lin et al. (2009) initialize β with an upper-bound obtained by optimizing a convex sub-expression of the full objective. Our preliminary experiments confirmed that this method yielded significantly better optima than random initializations. Therefore, we adopted this choice for initialization. As for the main optimization task, we have used the Spectral Projected Gradient (SPG) algorithm (Varadhan and Gilbert, 2009), a gradient-based method that allows for simple box constraints.

6 BENCHMARK ON SIMULATED DATA

In this section, we evaluate the empirical performance of our proposed EF approximations via comparison to the ET (Lin et al., 2009) and glasso (Friedman et al., 2008) algorithms on synthetic Gaussian and non-Gaussian data. We use the glasso implementation from the R-package *huge* (Zhao et al., 2012). The glasso regularization term is obtained via Stability Approach to Regularization Selection (StARS) (Liu et al., 2010), a criterion based on variability of the graphs estimated by overlapping subsamplings. We employ this criterion, since it achieves the best

performance in our simulations. For the ET and EF approaches we use Gaussian copula or Student's t-copula potentials and optimize the corresponding objective via SPG. For the EF-cuts method, we consider the first 50 minimum-weight cuts.

6.1 RESULTS ON GAUSSIAN MRF DATA

We first aim at confirming that the EF model achieves comparable performance to state-of-the-art methods for MRF structure learning. To this end, we generated Gaussian MRF data following the procedure described in (Lin et al., 2009). The off-diagonal entries of the precision matrix $\Omega = \Sigma^{-1}$ are sampled from $\pm(0.1 + 0.2|n|)$, where n is drawn from $\mathcal{N} \sim (0, 1)$. The diagonal entries are selected via Gershgorin's circle theorem to ensure that the matrix is positive definite. Given $\Omega = \Sigma^{-1}$, data can be easily sampled from a multivariate Gaussian distribution $\mathcal{N} \sim (\mathbf{0}, \Sigma)$.

We first generate random connected graphs of $d = 25$ nodes with an average of 2 neighbours/node. For a given graph, we draw 500 samples from the corresponding GMRF distribution and then compare the ability of different methods to retrieve the graph structure when a different sample size is available. Performance metrics for this setting, obtained from 100 repetitions, are reported in Figure 2A, while the average runtime for each method is given in Table 1. We can see that the EF- λ and EF-cuts approaches have similar accuracy as the ET, as the corresponding Hamming distances to the ground truth (i.e. number of misclassified edges) are on the same level. Notably, the number of false positive edges predicted by the EF- λ method is zero in most cases. Thus, precision is always very close to one. As a trade-off, recall is limited, especially for lower sample sizes. When 500 samples are available, recall reaches levels comparable to the baseline methods. The EF-cuts method performs very similar to the ET, while exhibiting a much higher runtime. The reported runtimes for EF- λ and glasso correspond to a complete run with 32 λ -values. The runtime for glasso is not dependent on the sample size and is mostly consumed for choosing the optimal λ . On the other hand, the runtime for EF- λ increases with sample size. However, we argue that the added runtime constitutes a reasonable trade-off for achieving superior structure learning performance.

Table 1: Average runtime (in seconds) for the experiments presented in Figure 2. For EF- λ and glasso the reported runtime corresponds to a complete run with 32 λ -values and choice of the optimal λ .

Sample Size:	25	50	100	250	500
ET	6	9	13	28	57
EF- λ	32	39	56	110	188
EF-cuts	1166	2088	3512	8151	14435
glasso	31	31	31	31	31

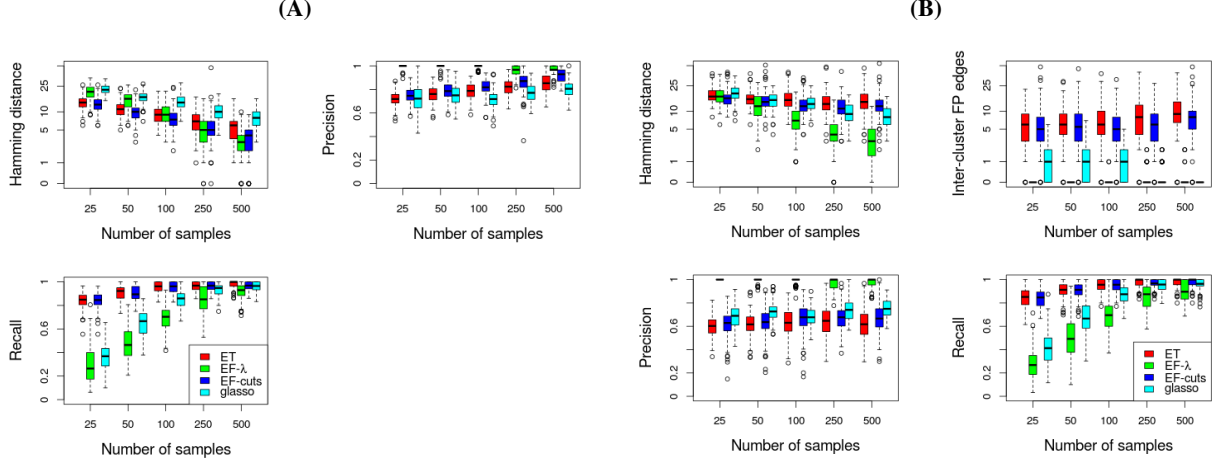


Figure 2: Comparison of the EF- λ , EF-cuts, ET and glasso algorithms on recovering the structure of (A) connected (B) disconnected sparse GMRFs from different sample sizes. Simulated graphs comprise 25 nodes with 2 neighbours/node on average. The boxplots contain results from 100 repetitions.

In a next step, we evaluated the performance of the EF model in a situation where the data is drawn from a Gaussian MRF with multiple connected components. Therefore, we generated data from GMRFs with no restriction on the number of connected components. Again, each graph comprises $d = 25$ nodes with an average of 2 neighbours/node. Performance metrics for this setting, obtained from 100 repetitions, are reported in **Figure 2B**. We can observe that the EF- λ approach outperforms the other three in terms of accuracy, as it achieves the lowest Hamming distance. As in the one-component setting, the number of false positive edges predicted by this method is zero in most cases. Thus, there are no inter-cluster false positive edges (i.e. edges that are falsely predicted to connect nodes belonging to different clusters) and precision is always very close to one. The recall achieved is inferior to the other methods. However, as the sample size grows, recall also reaches competitive levels. Again in this setting, the EF-cuts approach performs very similar to the original ET method.

We have seen that the EF-cuts method performs very similar to the original ET method, but exhibits much higher runtimes. On the other hand, the EF- λ heuristic performs very well for both connected and disconnected MRFs and is additionally faster and more generic than the EF-cuts. Thus, we only include EF- λ in the next simulations and refer to it as simply EF.

6.2 RESULTS ON NON-GAUSSIAN MRF DATA

Here we explore the ability to learn the structure of MRFs with non-Gaussian potentials. The EF, as well as the ET approach, are applicable for arbitrary potentials and are, therefore, expected to well adapt to this situation.

We now perform simulations for a Markov network whose data dependencies are no longer Gaussian. More specifi-

cally, we generate random graphs consisting of 25 nodes that are organized in small cliques of size 3 or 4. For each clique we draw data samples of pseudo-observations (Kojadinovic and Yan, 2010) from a Student’s t-copula with 1 degree of freedom. The dependencies among random variables in each clique are clearly non-Gaussian. Subsequently, we apply the Gaussian quantile function to the pseudo-observations of each random variable and, thereby, we obtain data that is marginally normally distributed. In this setting, we compare the EF approach to the ET, glasso and, additionally, to the non-paranormal model (npn) of Liu et al. (2009). The latter utilizes Gaussian copulas for structure learning. Its implementation is also available via the R-package *huge*.

The results of 100 simulations are summarized in the boxplots of **Figure 3**. The Hamming distances produced by the EF approach are considerably smaller than those produced by competing approaches. Moreover, no false positive edges are predicted by the EF method. Precision and also recall are very high. In contrast, the glasso and non-paranormal methods, that assume Gaussian dependency structures, achieve limited recall. The ET method produces higher Hamming distances and also low precision, since it introduces false positive edges that connect the cliques. Note that the Hamming distance for this method is almost equal to the number of inter-cluster false positive edges. In such a setting, the EF approach performs significantly better than all alternative methods since it naturally deals with t-copula dependencies and disconnected MRF topologies.

6.3 A HIGH-DIMENSIONAL SETTING WITH VERY LOW SAMPLE SIZE

Here, we explore structure learning on the basis of an extremely low number of samples from a comparably high dimensional MRF. This situation commonly arises in many

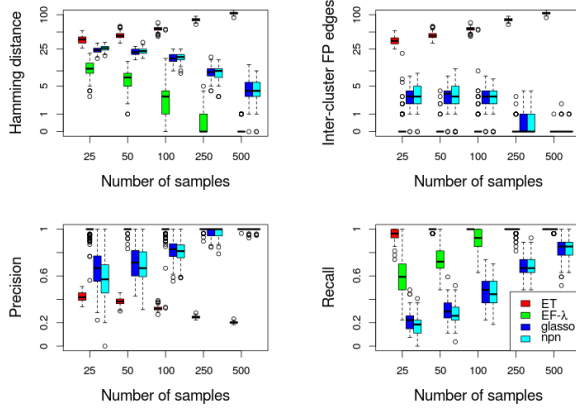


Figure 3: Comparison of the EF, ET, glasso and non-paranormal algorithms on recovering the structure of sparse MRFs with Student’s t -copula ($df = 1$) potentials. Simulated graphs comprise 25 nodes organized in small cliques of size 3 or 4. The boxplots contain results from 100 repetitions.

real world applications, as for instance in biology where typically only few observations are available. In this situation, we do not expect to comprehensively recover the underlying MRF structure. Instead, we aim to maximize the number of recovered true MRF edges at high precision, i.e. without accumulating false positive relationships. Therefore, we generate 50 data samples from an 80-dimensional GMRF, where each node has on average 3 neighbours. The ROC curves in **Figure 4** compare the performance of the EF and glasso approaches. We can see that, for very low sample sizes, the EF method recovers almost a double number of edges at a tolerance level of 1% FDR. In **Table 2** we present the average runtime for EF and glasso when run with a single λ value.

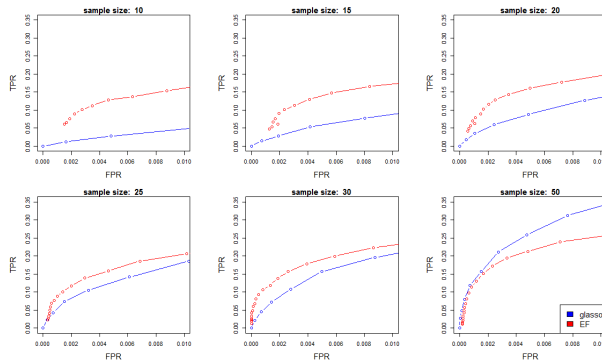


Figure 4: Comparison of the EF and glasso algorithms in a high-dimensional setting (80-node graph) with very low sample size. ROC curves for different numbers of available data replicates are presented, averaged over 100 repetitions. The curves are truncated at a tolerance level of 1% FDR.

Table 2: Average runtime (in seconds) for the simulations presented in **Figure 4**. Runtime is averaged over repetitions and λ values.

Sample Size:	10	15	20	25	30	50
EF- λ	107	153	163	182	185	248
glasso	< 1	< 1	< 1	< 1	< 1	< 1

7 RESULTS ON MICROARRAY DATA

Here we demonstrate the performance of the EF approach on a microarray dataset (Wille et al., 2004) from the isoprenoid biosynthesis pathways in *Arabidopsis thaliana*. Expression levels of 39 genes (variables) are quantified under $n = 118$ conditions (observations). EF is evaluated via comparison to glasso (Friedman et al., 2008), the state-of-the-art algorithm for learning the structure of continuous MRFs. For the EF analysis, we used the Gaussian, Gumbel, Clayton, Frank and Student’s t copula as candidate parametric families. A summary of the copula selection results is presented in **Table 4**, where we can observe that a variety of different dependency types is present.

For both methods, a decreasing sequence of 40 λ -values was used. The optimal regularization parameter λ for EF was obtained via eBIC (Foygel and Drton, 2010), resulting in a sparse MRF whose graph structure is depicted in **Figure 5A**. On the contrary, the use of information criteria (eBIC, StARS (Liu et al., 2010)) for glasso yielded very dense networks, as depicted in **Figure 5B**. In order to additionally compare both approaches with respect to results at similar sparsity levels, we also selected the glasso graph with the smallest Hamming distance with respect to the graph learned via EF. To evaluate the performance of the algorithms, we used a 5-fold cross validation setting and evaluated the best-fitting model on the basis of the average per-sample held-out log-likelihood. Results are shown in **Table 3** and demonstrate that the MRF learned via EF has better cross validation performance. Besides the performance advantage, we note that the sparse structure of EF model selection enables straightforward interpretation and further hypothesis generation by domain experts.

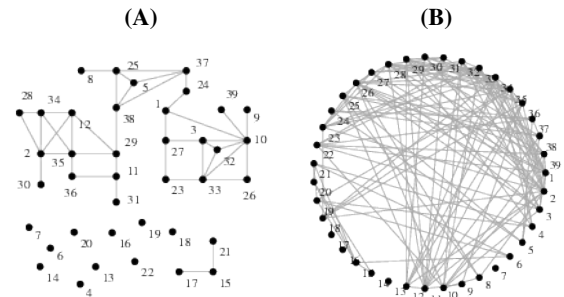


Figure 5: Optimal MRF graph structure recovered via (A) EF, (B) glasso for the microarray data. The numbering scheme legend is provided as Supplementary Material.

Table 3: Average per-sample held-out log-likelihood for the microarray data.

	Log-likelihood	Std. error
EF- λ	9.694	0.526
glasso (StARS)	8.522	0.418
glasso (sparse)	8.995	0.455

8 RESULTS ON IMMUNE CELL PERTURBATION DATA

Finally we apply the EF model to study the occurrence of MRFs with multiple connected components in a proteomics setting. Specifically, we analyze mass cytometry data from human peripheral blood mononuclear cells (PBMC), essentially representing all immune cells residing in the blood stream (Bodenmiller et al., 2012). Mass cytometry allows for proteomic profiling of molecular signaling events at single-cell resolution. The considered publicly available dataset recapitulates the response of PBMC populations to various molecular stimuli under several different pharmacological interventions. Signaling response has been measured by quantifying 14 phosphorylation sites (variables). For each intervention and cell type, 96 conditions were considered, where a condition consisted of an intervention strength setting and a specific stimulus.

Here we present results for interventions with the drug dasatinib. Again we observe the occurrence of a variety of non-Gaussian dependencies in this real world dataset (Table 4). We evaluate the performance of EF by comparing it to glasso, as we did for the microarray data. The average held-out log-likelihood per dataset is reported in the boxplots of Figure 6A. Different PBMC datasets are grouped together according to the stimulus used in each experiment. We can see that EF achieves constantly superior performance. Furthermore, in Figure 6B, separate histograms of the number of connected components for each stimulus are presented. For specific stimuli, MRF topologies with multiple components are common, reflecting the molecular impact of the intervention on the respective cellular signaling event. The EF approach is able to adapt to and recover underlying disconnected topologies even in the presence of unusual dependencies and, thus, we expect this approach to enable the probabilistic characterization of cellular signaling events and, thus, to enable molecular insights of possibly pathologically altered responses and to generate hypotheses for clinical interventions.

9 DISCUSSION

We have introduced the ensemble-of-forests model to approximate structure learning for MRFs with arbitrary potentials and connected components. Additionally, we have

Table 4: Frequencies of selected copula families during the analysis of plant microarray and PBMC mass cytometry data.

	Gumbel	Frank	Clayton	Gaussian	t (df=1)
Micro.	0.28	0.06	0.13	0.51	0.02
PBMC	0.20	0.06	0.35	0.23	0.16

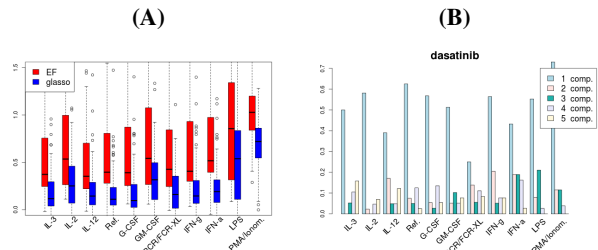


Figure 6: (A) Comparison of the EF and glasso algorithms. Boxplots of average held-out log-likelihood for different cell-type / stimulus combinations. (B) Histograms of the number of MRF connected components predicted by EF when applied to PBMC mass cytometry data. Separate histograms are given for each stimulus, indicated on the x-axis. Frequencies on the y-axis are normalized to sum up to 1 for each stimulus.

presented two approximate inference techniques for this model and compared their structure learning performance with state-of-the-art methods on a comprehensive set of synthetic data.

ET and EF models are appealing structure learning approaches when unusual MRF potentials are to be expected. Indeed, our simulation results confirm that the EF method can accurately reconstruct non-Gaussian dependencies that are a priori accounted for.

Disconnected dependency structures frequently arise in real world applications. However, the ET model is conceptually not able to handle such cases. We have extended the ET to the EF model to the end of accommodating multiple-component situations. Our simulation results confirm that we are able to faithfully recover MRF topologies with one as well as with multiple connected components. The study of the plant microarray and PBMC mass cytometry data furthermore confirms the ubiquitous occurrence of the multiple-component situation in cell biology and further emphasizes the need for structure learning approaches that are able to deal with this situation.

We also assessed how the EF model performs for limited sample size, again a typical case for real world applications. Our approach seems ideal for low-sample situations, where we aim to maximize the number of recovered true MRF edges at high precision.

In summary, we expect the EF model to enable MRF structure learning for many real world applications since this approach naturally deals with low sample size, unusual dependency types and disconnected dependency topologies.

References

- Banerjee, O., Ghaoui, L. E., d'Aspremont, A., and Sotgiu, G. (2006). Convex Optimization Techniques for Fitting Sparse Gaussian Graphical Models. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 89–96. ACM.
- Berkes, P., Wood, F., and Pillow, J. W. (2008). Characterizing neural dependencies with copula models. In *Advances in Neural Information Processing Systems*, pages 129–136.
- Bodenmiller, B., Zunder, E. R., Finck, R., Chen, T. J., Savig, E. S., Bruggner, R. V., Simonds, E. F., Bendall, S. C., Sachs, K., Krutzik, P. O., and Nolan, G. P. (2012). Multiplexed mass cytometry profiling of cellular states perturbed by small-molecule regulators. *Nature biotechnology*, 30(9):858–867.
- Elidan, G. (2013). Copulas in machine learning. In *Copulae in Mathematical and Quantitative Finance*, Lecture Notes in Statistics, pages 39–60. Springer Berlin Heidelberg.
- Embrechts, P., Lindskog, F., and McNeil, A. (2003). Modelling dependence with copulas and applications to risk management. *Handbook of heavy tailed distributions in finance*, 8(1):329–384.
- Foygel, R. and Drton, M. (2010). Extended Bayesian Information Criteria for Gaussian Graphical Models. In *Advances in Neural Information Processing Systems 23*, pages 604–612.
- Friedman, J., Hastie, T., and Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441.
- Johnstone, R. W., Frew, A. J., and Smyth, M. J. (2008). The TRAIL apoptotic pathway in cancer onset, progression and therapy. *Nature Reviews Cancer*, 8(10):782–798.
- Kirshner, S. (2008). Learning with Tree-Averaged Densities and Distributions. In *Advances in Neural Information Processing Systems*, pages 761–768.
- Kojadinovic, I. and Yan, J. (2010). Modeling multivariate distributions with continuous margins using the copula R package. *Journal of Statistical Software*, 34(9):1–20.
- Koller, D. and Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press.
- Lin, Y., Zhu, S., Lee, D. D., and Taskar, B. (2009). Learning Sparse Markov Network Structure via Ensemble-of-Trees Models. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, pages 360–367.
- Liu, H., Lafferty, J., and Wasserman, L. (2009). The Nonparanormal: Semiparametric Estimation of High Dimensional Undirected Graphs. *The Journal of Machine Learning Research*, 10:2295–2328.
- Liu, H., Roeder, K., and Wasserman, L. (2010). Stability Approach to Regularization Selection (StARS) for High Dimensional Graphical Models. In *Advances in Neural Information Processing Systems 23*, pages 1432–1440.
- Meilă, M. and Jaakkola, T. (2006). Tractable Bayesian learning of tree belief networks. *Statistics and Computing*, 16(1):77–92.
- Nelsen, R. B. (1999). *An introduction to copulas*. Springer.
- Trivedi, P. K. and Zimmer, D. M. (2005). Copula Modeling: An Introduction for Practitioners. *Foundations and Trends in Econometrics*, 1(1):1–111.
- Varadhan, R. and Gilbert, P. (2009). Bb: An R Package for Solving a Large System of Nonlinear Equations and for Optimizing a High-Dimensional Nonlinear Objective Function. *Journal of Statistical Software*, 32(4):1–26.
- Vazirani, V. and Yannakakis, M. (1992). Suboptimal cuts: Their enumeration, weight and number. In *Automata, Languages and Programming*, pages 366–377. Springer.
- Wille, A., Zimmermann, P., Vranova, E., Furholz, A., Laule, O., Bleuler, S., Hennig, L., Prelic, A., von Rohr, P., Thiele, L., Zitzler, E., Gruissem, W., and Buhlmann, P. (2004). Sparse graphical Gaussian modeling of the isoprenoid gene network in Arabidopsis thaliana. *Genome Biol.*, 5(11):R92.
- Zhao, T., Liu, H., Roeder, K., Lafferty, J., and Wasserman, L. (2012). The huge Package for High-dimensional Undirected Graph Estimation in R. *The Journal of Machine Learning Research*, 98888:1059–1062.

CAN(PLAN)+: Extending the Operational Semantics of the BDI Architecture to deal with Uncertain Information

Kim Bauters and Weiru Liu and Jun Hong
Queen's University Belfast
Belfast, United Kingdom

Carles Sierra and Lluís Godo
IIIA, CSIC, Bellaterra, Spain
Queen's University Belfast, Belfast, United Kingdom

Abstract

The BDI architecture, where agents are modelled based on their beliefs, desires and intentions, provides a practical approach to develop large scale systems. However, it is not well suited to model complex Supervisory Control And Data Acquisition (SCADA) systems pervaded by uncertainty. In this paper we address this issue by extending the operational semantics of CAN(PLAN) into CAN(PLAN)+. We start by modelling the beliefs of an agent as a set of epistemic states where each state, possibly using a different representation, models part of the agent's beliefs. These epistemic states are stratified to make them commensurable and to reason about the uncertain beliefs of the agent. The syntax and semantics of a BDI agent are extended accordingly and we identify fragments with computationally efficient semantics. Finally, we examine how primitive actions are affected by uncertainty and we define an appropriate form of lookahead planning.

1 INTRODUCTION

SCADA (Supervisory Control And Data Acquisition) systems are known for their large scale processes in a wide variety of domains, including production processes [Zhi et al., 2000] and energy and transportation systems [Boyer, 2009]. One way of modelling such systems is by means of the BDI architecture [Bratman, 1987] which allows us to decompose a complex system into a set of autonomous and interacting agents, where an agent is defined by its (B)eliefs, (D)esires and (I)ntentions. Agent-based programming languages based on the BDI framework have been proposed [Ingrand et al., 1992, Rao, 1996, Dastani, 2008] and have been used to some extent to model SCADA systems (e.g. [McArthur et al., 2007]).

However, current BDI implementations are not well-suited to model the next generation of complex SCADA systems.

The reason for this is two-fold. On the one hand, current BDI implementations are not able to deal with uncertainty associated with the beliefs of an agent (e.g. due to noisy sensing) or the uncertain effects of actions (e.g. due to actuator malfunctions). This limits the ability of a BDI agent to react in a satisfactory way in an uncertain environment. On the other hand, and closely related, is that most BDI implementations do not provide any mechanisms for lookahead planning to guide (part of) the BDI execution in this uncertain setting.

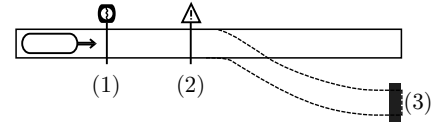


Figure 1: Scenario for a train agent with an unreliable signal (1), a dangerous junction (2) and a goal station (3).

To illustrate these issues, consider the running example in Figure 1. A train agent is moving along a track with a signal (1). The signal, which is green or orange, informs the agent if it violates the safe distance (with uncertainty due to e.g. mist, conflicting signals ...). Once the agent has passed the signal, the agent decides on how to approach the junction (2). The speed of the train is not known exactly, yet the agent needs to decide whether it wants to keep speeding (as it is running late) or slow down (resp. 75% and 50% chance of reaching the junction in time). Once at the junction, the action to take the junction only has a 30% chance of succeeding when speeding (e.g. due to derailment). Otherwise, the junction can safely be taken. For simplicity, the station is reached on time only when the junction is safely taken. Clearly, an agent should be able to reason about the uncertainty and be able to plan ahead, e.g. foresee that slowing down is the best action.

Not a lot of work in the literature on BDI tackled the problem of representing, and reasoning about, uncertain information. A notable exception is the recent work in [Chen et al., 2013], which incorporates uncertain perceptions in the epistemic state of an agent after which

it is mapped to a classical belief base, thus ignoring the other information. The work on graded BDI systems [Casali et al., 2005, Casali et al., 2011] similarly discusses how uncertainty can pervade the beliefs, desires and intentions. However, the graded BDI framework is mainly of theoretical interest and has not led to actual implementations, contrary to how AgentSpeak and CAN have helped to advance the state-of-the-art in BDI implementations.

Planning in a BDI agent, where the agent reflects on its actions before executing them, has been considered in numerous works. While the BDI model does not prevent planning, most BDI implementations resort to simple plan selection strategies to avoid the computational cost associated with declarative planning. This prevents them from acting optimally when needed, e.g. when important resources are consumed during the execution of plans. A formal approach to planning in BDI, called CANPLAN, was presented in [Sardiña et al., 2006]. CANPLAN is based on the Conceptual Agent Notation (CAN) [Winikoff et al., 2002], a high-level agent language in the spirit of BDI [Rao and Georgeff, 1991]. It is closely related to AgentSpeak [Rao, 1996] but allows for declarative goals alongside procedural steps (i.e. we can state *what* we want to achieve, not just *how* to achieve it). CANPLAN extends this work by introducing a $Plan(\cdot)$ action, making planning on-demand an integral part of the BDI framework. Nevertheless, none of the approaches to BDI address the issues that arise when dealing with actions with uncertain effects, or uncertain beliefs in general.

In this paper we propose the CAN+ and CANPLAN+ frameworks, which extend CAN and CANPLAN to provide formal approaches for dealing with uncertain beliefs and (planning for) actions with uncertain effects. The beliefs of an agent are modelled as a set of epistemic states, with each *local* epistemic state representing a distinct part of the beliefs held by the agent. Each epistemic state can deal with a different form of uncertainty (e.g. possibilities or infinitesimal probabilities) and includes its own revision strategy. Such a set of local epistemic states will be called a *Global Uncertain Belief set* (GUB) and allows the agent to reason about different forms of uncertainty in a uniform way, as long as these can be expressed using epistemic states that are equivalent to Definition 1. This is achieved in two steps. Firstly, a stratification of each local epistemic state allows for commensurability, along with the ability to reason about the uncertain beliefs. In other words: it enables an agent to reason about those beliefs it currently does not assume to be true (in the sense of beliefs in classical logics). Nevertheless, an agent commonly still considers some outcomes to be more plausible than others. The agent thus gains the ability to reflect on its own uncertainty. Secondly, an agent will be able to revise a GUB directly, with the GUB ensuring that only the information relevant to a specific local epistemic state is used to revise it. This idea of

a GUB will be introduced in the CAN framework to obtain CAN+. CANPLAN+ further extends upon it by adding the ability to execute and plan for non-deterministic actions, all while dealing with uncertain beliefs.

The remainder of this paper is organised as follows. Some preliminary notions are discussed in Section 2. We explore how we can efficiently model and reason about uncertain beliefs in Section 3, where we introduce the idea of epistemic states and how they can be applied in a BDI setting. In Section 4 we extend CAN to enable us to deal with uncertain beliefs, while uncertain actions and planning under uncertainty are addressed in Section 5. Related work is discussed in Section 6 and conclusions are drawn in Section 7.

2 PRELIMINARIES

An agent in the BDI framework is defined by its beliefs, desires and intentions. The beliefs encode the agent's understanding of the environment, the desires are those goals that an agent would like to accomplish and the intentions those desires that the agent has chosen to act upon.

CAN, and its extension CANPLAN, formalise the behaviour of a classical BDI agent, which is defined by a belief base \mathcal{B} and a plan library Π . The belief base of an agent is a set of formulas over some logical language that supports entailment (i.e. $\mathcal{B} \models b$, b a belief), belief addition and belief deletion (resp. $\mathcal{B} \cup \{b\}$ and $\mathcal{B} \setminus \{b\}$). The plan library is a set of plans of the form $e : \psi \leftarrow P$ where e is an event, ψ is the context and P is a plan body. Events can either be external (i.e. from the environment in which the agent is operating) or internal (i.e. sub-goals that the agent itself tries to accomplish). The plan body P is *applicable* to handle the event e when $\mathcal{B} \models \psi$, i.e. the context evaluates to true. The event and context differ in that the context is lazily evaluated; it is checked right before the execution of the plan body. The language used in the plan body P is defined in Backus-Naur Form (BNF) as:

$$P ::= nil \mid +b \mid -b \mid act \mid ?\phi \mid !e \mid P_1; P_2 \mid P_1 \parallel P_2 \mid P_1 \triangleright P_2 \mid (|\Delta|) \mid Goal(\phi_s, P, \phi_f) \mid Plan(P)$$

with nil an empty or completed program, $+b$ and $-b$ belief addition and deletion, act a primitive action, $?\phi$ a test for ϕ in the belief base, and $!e$ a subgoal, i.e. an (internal) event. Actions, tests and subgoals can fail, e.g. when the preconditions are not met. Composition is possible through $P_1; P_2$ for sequencing, $P_1 \parallel P_2$ for parallelism (i.e. a non-deterministic ordering) and $P_1 \triangleright P_2$ to execute P_2 only on failure of P_1 . $(|\Delta|)$ is used to denote a set of guarded plans, with Δ of the form $\psi_1 : P_1, \dots, \psi_n : P_n$, which intuitively states that the plan body P_i is *guarded* by the context ψ_i , i.e. the context needs to be true to execute the plan body. The plan form $Goal(\phi_s, P, \phi_f)$ is a distinguishing feature of CAN that allows to model both declarative and procedural goals. It states that we should achieve the (declarative)

goal ϕ_s using the (procedural) plan P , where the goal fails if ϕ_f becomes true during the execution. CANPLAN furthermore introduces the $Plan(P)$ construct, which is used for offline lookahead planning. This construct will be discussed in more detail in Section 5.

The operational semantics of CANPLAN are defined in terms of configurations. A basic configuration is a tuple $\langle \mathcal{B}, \mathcal{A}, P \rangle$ with \mathcal{B} a belief base, \mathcal{A} the sequence of primitive actions that have been executed so far and P the remainder of the plan body to be executed (i.e. the current intention). An agent (configuration) is a tuple $\langle \mathcal{N}, \mathcal{D}, \Pi, \mathcal{B}, \mathcal{A}, \Gamma \rangle$ with \mathcal{N} the name of the agent, \mathcal{D} the action description library, Π the plan library, Γ the set of current intentions of the agent and \mathcal{B} and \mathcal{A} as before. For each action act the action description library contains a rule of the form $act : \psi \leftarrow \phi^-; \phi^+$. We have that ψ is the precondition, while ϕ^- and ϕ^+ denote respectively a delete and add set of belief atoms, i.e. propositional atoms.

A *transition relation* \longrightarrow on (both types of) configurations is defined by a set of derivation rules. A transition $C \longrightarrow C'$ denotes a single step execution from C yielding C' . We write $C \longrightarrow$ to state there exists a C' such that $C \longrightarrow C'$ and $C \not\longrightarrow$ otherwise. We use $\xrightarrow{*}$ to denote the transitive closure over \longrightarrow . A *derivation rule* consists of a (possibly empty) set of premises p_i and a single transition conclusion c . Such a derivation rule is denoted as

$$\frac{p_1 \quad p_2 \quad \dots \quad p_n}{c} l$$

with l a label attached to the derivation rule for easy reference. Transitions over basic configurations (resp. agent configurations) define what it means to execute a single intention (resp. the agent as a whole). For example, the transition for belief addition and a primitive action are:

$$\frac{\langle \mathcal{B}, \mathcal{A}, +b \rangle \longrightarrow \langle \mathcal{B} \cup \{b\}, \mathcal{A}, nil \rangle +b}{(a : \psi \leftarrow \phi^-; \phi^+) \in \mathcal{D} \quad a\theta = act \quad \mathcal{B} \models \psi\theta \quad \langle \mathcal{B}, \mathcal{A}, act \rangle \longrightarrow \langle (\mathcal{B} \setminus \phi^- \theta) \cup \phi^+ \theta, \mathcal{A} \cdot act, nil \rangle} act$$

where the latter states that when the unified precondition $\psi\theta$ is true in the belief base \mathcal{B} , the effect of the action is the application of the add and delete atom lists to the belief base. We refer the reader to [Sardiña and Padgham, 2011] for a full overview of the semantics of CANPLAN.

Finally, a *preorder* \leq_A defined on any set A is a reflexive and transitive relation over $A \times A$. We say that \leq_A is *total* iff for all $a, b \in A$ we have that either $a \leq_A b$ or $b \leq_A a$. A strict order $<_A$ and an indifference relation $=_A$ can conventionally be induced from \leq_A .

3 MODELLING AND REASONING ABOUT UNCERTAIN BELIEFS

As discussed in Section 2, a belief base in CAN is defined over a logic for which operations are available to add,

delete and entail beliefs. This classical setting allows for an easy approach to belief revision. However in this paper we are concerned with the modelling of, and reasoning over, uncertain information. To deal with uncertainty we will need more elaborate ways to both represent the beliefs and to revise the beliefs when new information becomes available. To this end, we will use epistemic states instead of a belief base as in CAN.

3.1 MODELLING UNCERTAIN BELIEFS AS EPISTEMIC STATES

To define epistemic states, we first start by considering a finite set \mathcal{At} of propositional atoms. For a set of atoms $A \subseteq \mathcal{At}$ we define the set of literals that can be constructed using the atoms in A as $lit(A) = \{a \mid a \in A\} \cup \{\neg a \mid a \in A\}$. A proposition ϕ is defined in BNF as $\phi ::= a \mid \neg a \mid (\phi_1 \wedge \phi_2) \mid (\phi_1 \vee \phi_2)$, i.e. all propositions are in Negation Normal Form (NNF). This does not affect the expressiveness of our language as arbitrary formulas can be efficiently converted into NNF. It will, however, make the definition of our semantics easier. We will denote this language as \mathcal{L} . Now we introduce the concept of an epistemic state:

Definition 1. (from [Ma and Liu, 2011]) Let Ω be a set of possible worlds. An epistemic state is a mapping $\Phi : \Omega \rightarrow \mathbb{Z} \cup \{-\infty, +\infty\}$.

An epistemic state will be used to represent the mental state of an agent, where the value $\Phi(\omega)$ associated with a possible world ω , called the *weight* of ω , is understood as the degree of belief in the possible world ω . Throughout the paper we will denote epistemic states using capital Greek letters. For $\omega, \omega' \in \Omega$ and $\Phi(\omega) > \Phi(\omega')$ the intuition is that ω is more plausible than ω' . Two epistemic states Φ and Ψ are *semantically equivalent* iff $\exists k \in \mathbb{Z} \cdot \forall \omega \in \Omega : \Phi(\omega) = \Psi(\omega) + k$, i.e. the value associated with the possible worlds only has a relative meaning. In the remainder of this paper we assume that epistemic states have 2^A as their domain with $A \subseteq \mathcal{At}$. The *strength* of preference on a propositional formula ϕ is defined as $\Phi(\phi) - \Phi(\neg\phi)$ with $\Phi(\phi) = \max_{\omega \models \phi} (\Phi(\omega))$. We use \max_Φ to denote $\max_{\omega \in \Omega} (\Phi(\omega)) - \min_{\omega \in \Omega} (\Phi(\omega)) + 1$, i.e. the weight stronger than any of the strengths associated with information in Φ , and $\min_\Phi = 1 - \max_\Phi$.¹ The values \max_Φ and \min_Φ are only needed in Section 4 when considering belief additions and deletions as in CAN.

Before we provide an example, it is important to clarify that the definition of an epistemic state from Definition 1 allows for the construction of a general framework. Indeed, this definition does not impose any restrictions on the values associated with the possible worlds, other than that they are weights. As such, it is the

¹These values only change when the epistemic state is revised and can be computed as a by-product of revision.

most general way in which we can talk about an epistemic state, regardless of the actual representation. Other representations for epistemic states, which attach more specific meaning to the values, have been shown to be equivalent to the one from Definition 1. For example, Definition 1 induces an Ordinal Conditional Function (OCF) [Spohn, 1988]², which in turn can be transformed into other representations, e.g. those based on infinitesimal probabilities [Darwiche and Goldszmidt, 1994] and possibility theory [Dubois and Prade, 1995]. The representation from Definition 1 can thus be *instantiated* using any of the other representations to best suit the nature of the uncertainty. After developing all main concepts, we will show in Section 4 that this will allow us to work with these different forms of uncertainty in a uniform way.

We now give an example of such an epistemic state.

Example 1. Consider a signal that can be (o)range or (g)reen (but never both on). When the signal is orange it usually indicates that the agent is about to violate the safe distance (sd). The agent believes that the light is green and that there is still a safe distance with the train in front. Even when the signal would turn out not to be green, the agent still believes that there would be a safe distance with the train in front of it. An epistemic state Φ could be:

$$\begin{aligned}\Phi(\{o, g, sd\}) &= -\infty & \Phi(\{\neg o, g, sd\}) &= 10 \\ \Phi(\{o, g, \neg sd\}) &= -\infty & \Phi(\{\neg o, g, \neg sd\}) &= -2 \\ \Phi(\{o, \neg g, sd\}) &= 7 & \Phi(\{\neg o, \neg g, sd\}) &= 7 \\ \Phi(\{o, \neg g, \neg sd\}) &= 6 & \Phi(\{\neg o, \neg g, \neg sd\}) &= -2\end{aligned}$$

where $\max_{\Phi} = +\infty$ and $\min_{\Phi} = -\infty$. The weight associated with e.g. $\{o, g, sd\}$ means that we strongly disbelieve this possible world, while the weight associated with $\{\neg o, g, sd\}$ implies that we believe this possible world to be more plausible than any of the other possible worlds.

The belief set, i.e. the sentences that an agent is committed to believe, is commonly defined as the set that has all the most plausible worlds as its models.

Definition 2. (from [Ma and Liu, 2011]) Let Φ be an epistemic state. The belief set of Φ , denoted as $Bel(\Phi)$, is defined as $Bel(\Phi) = \phi$ with ϕ any propositional formula such that $Mod(\phi) = \min(\Omega, \leq_{\Phi})$. Here $Mod(\phi)$ is the set of models of ϕ and \leq_{Φ} is a total preorder relation over Ω such that $\omega \leq_{\Phi} \omega'$ iff $\Phi(\omega) \geq \Phi(\omega')$.

The model of the belief set thus only contains those possible worlds with the highest weight. In this paper we extend on this idea by also taking the other possible worlds into account. These other possible worlds constitute the uncertain information, i.e. they define the preferences the agent has over the outcomes that are currently not believed to be true.

²Compared to OCFs, the representation from Definition 1 avoids the normalisation step.

To clearly identify these preferences, irrespective of the actual values of these weights in different representations of the epistemic states, we consider a stratification of the set of possible worlds. The highest stratum (containing those possible worlds with the strongest associated belief) corresponds to the set of models of the belief set, i.e. that what the agent believes to be true. The other strata constitute the uncertain information, with information in a higher stratum believed/preferred over information in a lower stratum.

Definition 3. Let Φ be an epistemic state. The stratification λ of the domain Ω from Φ induced by the total preorder relation \leq_{Φ} is defined as:

$$\lambda(\omega) = \begin{cases} 1 & \omega \in \min(\Omega, \leq_{\Phi}) \\ n+1 & \omega \in \min(\Omega \setminus \{\omega \mid \lambda(\omega) \leq n\}, \leq_{\Phi}) \end{cases}$$

In Example 1 we obtain $\lambda(\{\neg o, g, sd\}) = 1$, $\lambda(\{o, \neg g, sd\}) = \lambda(\{\neg o, \neg g, sd\}) = 2$, etc. This idea of a stratification readily corresponds with the more common notion of a stratification over propositional formulas as any subset of possible worlds can trivially be represented by a single propositional formula.

Notice that the models of the belief set (see Definition 2) correspond to those possible worlds ω for which $\lambda(\omega) = 1$, i.e. information on all the other strata is ignored. Instead of simply ignoring this information, we want to make it possible for a BDI agent to reason about the preferences expressed throughout the stratification. To this end we extend the language \mathcal{L} with the connectives \geq and $>$. The intuition of $a \geq b$ (resp. $a > b$) is that the agent believes a to be at least as plausible as b (resp. a is strictly more plausible than b). These new connectives are taken to have the lowest precedence. The resulting language \mathcal{L}_{\geq} , or the *context language*, can be defined in BNF as:

$$\phi ::= a \mid \neg a \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \phi_1 \geq \phi_2 \mid \phi_1 > \phi_2$$

where formulas with connectives such as \rightarrow and \leftrightarrow can easily be transformed into logically equivalent statements in the language \mathcal{L}_{\geq} . Notice that this definition is the equivalent of NNF for propositional formulas. Any proposition using the connectives \neg , \wedge , \vee , \geq and $>$ can be turned into an equivalent formula in \mathcal{L}_{\geq} in the usual way and by rewriting $\neg(\psi_1 \geq \psi_2)$ as $(\psi_2 > \psi_1)$ and $\neg(\psi_1 > \psi_2)$ as $(\psi_2 \geq \psi_1)$. We assume that this has been done when needed throughout paper.

By extending the mapping λ we can define the semantics of \mathcal{L}_{\geq} over arbitrary formulas. We have:

$$\lambda(\phi) = \begin{cases} \min \{\lambda(\omega) \mid \omega \models \phi\} & \text{if } \phi \in \mathcal{L} \\ \lambda(pare(\phi)) & \text{otherwise} \end{cases}$$

with $\min(\emptyset) = \infty$. Before defining the function *pare*, we point out that λ is closely related to a possibility measure [Dubois et al., 1994] for propositional

formulas $\phi, \psi \in \mathcal{L}$. We readily establish some interesting properties such as $\lambda(\phi \vee \psi) = \min(\lambda(\phi), \lambda(\psi))$, $\lambda(\phi \wedge \psi) \geq \max(\lambda(\phi), \lambda(\psi))$, $\lambda(\top) = 1$, $\lambda(\perp) = \infty$ and $\min(\lambda(\phi), \lambda(\neg\phi)) = 1$.

When ϕ is not a propositional statement (i.e. $\phi \notin \mathcal{L}$), we need to pare down the formula until the formula is a classical propositional statement. This is done by:

$$\begin{aligned} \text{pare}(\phi \wedge \psi) &= \text{check}(\phi) \wedge \text{check}(\psi) \\ \text{pare}(\phi \vee \psi) &= \text{check}(\phi) \vee \text{check}(\psi) \\ \text{pare}(\phi \geq \psi) &= \begin{cases} \top & \text{if } \lambda(\neg\phi) \geq \lambda(\neg\psi) \\ \perp & \text{otherwise} \end{cases} \\ \text{check}(\phi) &= \begin{cases} \phi & \text{if } \phi \in \mathcal{L} \\ \text{pare}(\phi) & \text{otherwise} \end{cases} \end{aligned}$$

with $\text{pare}(\phi > \psi)$ equivalently defined as $\text{pare}(\phi \geq \psi)$. The intuition of paring down is straightforward: for each operand of the operators \wedge and \vee we verify whether it is an expression in the language \mathcal{L} (for which the λ -value can readily be determined). Otherwise, we need to further pare it down. When the operator is \geq or $>$, we define it as a plausibility ordering with an expression such as $\phi > \psi$ read as “ ϕ is more plausible than ψ ” or, alternatively, “we have less reason to believe $\neg\phi$ than $\neg\psi$ ”.³ Such an expression can always be evaluated to true or false, i.e. \top or \perp .

Finally, we can define when a formula ϕ is entailed.

Definition 4. Let Φ be an epistemic state and ϕ a formula in \mathcal{L}_{\geq} . We say that ϕ is entailed by Φ , written as $\Phi \models \phi$, if and only if $\lambda(\phi) < \lambda(\neg\phi)$.

Example 2. Consider λ of Φ from Example 1. We have:

$$\begin{aligned} \lambda(g \wedge sd) &= 1 & \lambda(o \wedge sd) &= 2 & \lambda(g \wedge \neg g) &= \infty \\ \lambda((o \vee g) > \neg sd) &= 1 & \lambda(g \geq o) &= 1 & \lambda(o \geq g) &= \infty \end{aligned}$$

For example, $\lambda(g \wedge sd) = 1$ since $\lambda\{\neg o, g, sd\} = 1$ and $\{\neg o, g, sd\} \models g \wedge sd$. An expression such as $(o \vee g) > \neg sd$, which is also believed to be true, states that the agent does not care about the colour of the light as long as the agent is not violating the safe distance.

Note that in Definition 4 it is insufficient to state that a formula is entailed when $\lambda(\psi) = 1$. Indeed, for $a \in \mathcal{At}$ we can have that $\lambda(a) = \lambda(\neg a) = 1$, which occurs when we are ignorant about the value of a . As such, we need to ensure that both expressions are mapped onto strictly distinct strata. This notion of entailment (assuming $\psi \in \mathcal{L}$) corresponds exactly to those formulas that can be derived from the belief base $\text{Bel}(\Phi)$.

Proposition 1. Let $\phi \in \mathcal{L}$ be a propositional formula, Φ an epistemic state with domain Ω and λ the stratification of Ω . We have that $\Phi \models \phi$ iff for all $\omega \in \Omega$ such that $\lambda(\omega) = 1$ we have that $\omega \models \phi$, i.e. $\text{Bel}(\Phi) \models \phi$.

³In terms of possibilistic theory: we want $N(\phi) \geq N(\psi)$, i.e. we want $\Pi(\neg\phi) \leq \Pi(\neg\psi)$ (with λ a reversed order).

3.2 SEMANTICS BASED ON LITERAL MAPPING

While the semantics we presented thus far allows us to reason about the uncertain information, they are computationally expensive for evaluating a context because they rely on an exponential structure. A tractable way to evaluate contexts can be obtained by restricting ourselves to a fragment of the language \mathcal{L}_{\geq} , allowing us to determine the truth of a context based on the λ -value associated with the constituent literals.

Example 3. Consider the stratification λ of Φ from Example 1. We have $\lambda(o) = 2$, $\lambda(\neg o) = 1$, $\lambda(g) = 1$, $\lambda(\neg g) = 2$, $\lambda(sd) = 1$ and $\lambda(\neg sd) = 3$.

Due to the way we defined λ over arbitrary formulas, we know that $\lambda(\phi \vee \psi)$ is decomposable, while $\lambda(\phi \wedge \psi)$ is not. Indeed, recall that we only have that $\lambda(\phi \wedge \psi) \geq \max(\lambda(\phi), \lambda(\psi))$. Therefore, we cannot allow \wedge in our restricted language. Furthermore, it affects our ability to verify whether for a given expression ψ we have that $\lambda(\psi) < \lambda(\neg\psi)$. Indeed, we can only allow disjunction as part of an operand of the operators \geq or $>$ as otherwise its negation would turn it into a conjunction, which we do not allow. As such, we obtain the fragment \mathcal{L}_{\geq} , defined in BNF as:

$$d ::= a \mid \neg a \mid d_1 \vee d_2 \quad \phi ::= a \mid \neg a \mid d_1 \geq d_2 \mid d_1 > d_2$$

Contexts in this language can easily be evaluated, once we have the λ -values of the literals $\text{lit}(A)$:

$$\begin{aligned} \lambda(\phi \vee \psi) &= \min(\lambda(\phi), \lambda(\psi)) \\ \lambda(\phi \geq \psi) &= \begin{cases} 1 & \lambda(\neg\phi) \geq \lambda(\neg\psi) \\ \infty & \text{otherwise} \end{cases} \end{aligned}$$

and equivalently for $\lambda(\phi > \psi)$. As before, we say that ϕ is true iff $\lambda(\phi) < \lambda(\neg\phi)$. Even though enforcing tractability carries a penalty in terms of the expressive power, we still retain a language that takes advantage of the new connectives we have introduced, thus allowing us to reason over the plausibility of statements.

3.3 EFFICIENTLY MODELLING ISOLATED UNCERTAIN BELIEFS

As a final step in the representation of uncertain beliefs for a BDI agent, we introduce the concept of a global uncertain belief set (GUB) which applies to both Section 3.1 and 3.2.

Definition 5. A global uncertain belief set \mathcal{G} is a set $\{\Phi_1, \dots, \Phi_n\}$ with each Φ_i an epistemic state over the domain $A_i \subseteq \mathcal{At}$ such that $\{A_1, \dots, A_n\}$ is a partition of \mathcal{At} .

Each local (or *isolated*) epistemic state models beliefs that are semantically related, e.g. the colour of the signal or the condition of the track, and that are governed by the same

form of uncertainty. A GUB then groups a set of such local epistemic states. A GUB is therefore a representation of the overall beliefs of an agent, yet it differs in three significant ways from a global epistemic state. First, it avoids the exponential representation of a global epistemic state by partitioning the beliefs. Second, it allows for a general framework where each local epistemic state can use a different representation. Third, it does not include a revision strategy (as each local epistemic state can have a distinct revision strategy), i.e. it is not itself an epistemic state.

Despite these differences, we can use a GUB to determine if a context ϕ is true according to the agent's collective beliefs. Intuitively, ϕ can be evaluated directly if it applies to a single local epistemic state Φ_i , i.e. we can verify whether $\Phi_i \models \phi$. Otherwise, we need to break ϕ apart up until the point where we can evaluate it directly. An expression can be split when the connective is either \wedge or \vee . Since both operands will either be true or false such a decomposition is trivial. However, this also implies that we cannot decompose \geq or $>$, since in this paper we require both operands to be from the same local epistemic state. Indeed, in general, stratifications of different formulas in different local epistemic states are incomparable due to the varying underlying structures. The problem of comparing the plausibilities of different local epistemic states is left for future work.

To formalise this intuition, we use $\mathcal{L}_{\geq}^{A_i}$ to denote the language \mathcal{L}_{\geq} limited to atoms $a \in A_i$, i.e. the language corresponding to the epistemic state Φ_i . A formula ϕ is broken apart by (*simp*)plifying it, which returns the evaluation of ϕ by evaluating the operands (or it returns \perp if the connective is \geq or $>$ and both operands are incomparable). We can then define $val_{GUB}(\phi)$ as:

$$val_{GUB}(\phi) = \begin{cases} \top & \text{if } \phi \in \mathcal{L}_{\geq}^{A_i}, \Phi_i \models \phi \\ \perp & \text{if } \phi \in \mathcal{L}_{\geq}^{A_i}, \Phi_i \not\models \phi \\ simp(\phi) & \text{otherwise} \end{cases}$$

$$simp(\phi \wedge \psi) = val_{GUB}(\phi) \wedge val_{GUB}(\psi)$$

$$simp(\phi \vee \psi) = val_{GUB}(\phi) \vee val_{GUB}(\psi)$$

$$simp(\phi \geq \psi) = \perp$$

and $simp(\phi > \psi)$ equivalently defined as $simp(\phi \geq \psi)$.

Definition 6. Let \mathcal{G} be a GUB and ϕ a formula in \mathcal{L}_{\geq} . We say that ϕ is entailed by \mathcal{G} , written as $\mathcal{G} \models \phi$, if and only if $val_{GUB}(\phi) \equiv \top$.

A visual representation of a GUB is given in Figure 2.

4 DEALING WITH UNCERTAIN BELIEFS IN A BDI AGENT

In the previous section we discussed how the beliefs of an agent can be represented as a set of local epistemic states. We also discussed how a GUB, and the underlying strat-

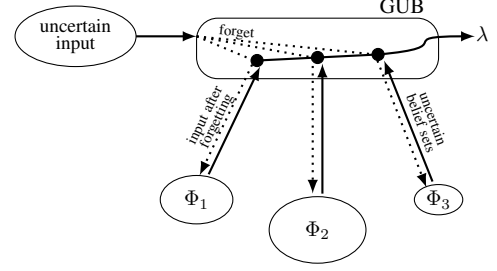


Figure 2: A GUB models the belief of an agent as a set of epistemic states Φ_i , each having its own representation (Definition 5). Commensurability is obtained by stratifying the possible worlds, with each stratum constituting an (uncertain) belief set (Definition 3). These stratifications can be combined to compute the λ -value of any arbitrary propositional formula. When new input is received, the local epistemic states are revised by ignoring (or forgetting) irrelevant information, as discussed in Section 4.

ification of the local epistemic states, can be used to ensure commensurability. In this section, we extend CAN to CAN+ by adding to it a GUB to represent uncertain beliefs and by extending its syntax so that a CAN+ agent can reason about its uncertain beliefs. After Definition 8, we introduce how a GUB can be revised directly, thus allowing an agent to revise its beliefs irrespective of the various forms of uncertainty that govern those beliefs. In our extension CAN+, a context ψ is taken to be a sentence from the language \mathcal{L}_{\geq} . We assume the language for a plan body to be defined as in CAN, where we will gradually modify the language throughout this section. First though, we redefine the concept of configurations in CAN+. Rather than considering a belief base to model the knowledge, we will thus use a GUB to represent the uncertain beliefs of the agent. We have:

Definition 7. A basic configuration is a tuple $\langle \mathcal{G}, \mathcal{A}, P \rangle$ with \mathcal{G} a GUB, \mathcal{A} the list of executed actions and P a plan body being executed (i.e. the current intention). An agent (configuration) is a tuple $\langle \mathcal{N}, \mathcal{D}, \Pi, \mathcal{G}, \mathcal{A}, \Gamma \rangle$ with \mathcal{N} the name of the agent, \mathcal{D} the action description library (defined in Section 5), Π the plan library, Γ the set of current intentions of the agent and \mathcal{G} and \mathcal{A} as before.

With the configurations redefined we can extend the first set of rules from CAN, i.e. the rule for a test goal ($? \phi$) and the rule for plan selection (*select*):

$$\frac{\mathcal{G} \models \phi \theta}{\langle \mathcal{G}, \mathcal{A}, ? \phi \rangle \longrightarrow \langle \mathcal{G}, \mathcal{A}, nil \rangle} ? \phi$$

$$\frac{\psi_i : P_i \in \Delta \quad \mathcal{G} \models \psi_i \theta}{\langle \mathcal{G}, \mathcal{A}, (|\Delta|) \rangle \longrightarrow \langle \mathcal{G}, \mathcal{A}, P_i \theta \triangleright (|\Delta| \setminus P_i) \rangle} select$$

We retain the notation as used in [Sardiña et al., 2006] to denote unification as e.g. $\phi \theta$, i.e. variables are dealt with in the customary way. The modified rules make clear that verifying whether a belief or context holds is now done against

the GUB. The language has implicitly been extended in both cases, since test goals and contexts can now include statements to reason over uncertain beliefs, i.e. $\phi, \psi_i \in \mathcal{L}_{\geq}$.

So far we have looked at how we can reason about the agent's (uncertain) beliefs, but we also want to revise these beliefs. When new input is presented (e.g. due to an internal belief change or the effects of an action), a naive approach would be to compute the global epistemic state as the Cartesian product of the local epistemic states, apply the input and then marginalise the outcome. However, such an approach is computationally too expensive. Instead, we will apply the input directly to the relevant epistemic states. First though, we define the notion of an *uncertain belief*.

Definition 8. Let ϕ be a sentence in the language $\mathcal{L}^{A_{in}}$ with $A_{in} \subseteq A$. Let $\mu \in (\mathbb{Z} \cup \{-\infty, +\infty\})$. We say that $b = (\phi, \mu)$ is an uncertain belief.

An input b , which is an uncertain belief, corresponds to a sequence of inputs $refine(b, \Phi_i)$ for any given local epistemic state $\Phi_i \in \mathcal{G}$. We have:

$$refine(b, \Phi_i) = \begin{cases} forget(b, \Phi_i) & A_{in} \cap A_i \neq \emptyset \\ \langle \rangle & \text{otherwise} \end{cases}$$

with $forget(b, \Phi_i)$ a sequence of inputs defined as $\langle (m', \mu) \mid b = (\phi, \mu), m \in Mod(\phi), m' = m \cap lit(A_i) \rangle$ and m' in (m', μ) treated as a conjunction of literals. When $A_{in} \subseteq A_i$ we could equivalently take $forget(b, \Phi_i) = \langle b \rangle$.

By $\mathcal{G} \circ b$ we denote that we want to revise the current beliefs of the agent with the input b , such that $\mathcal{G} \circ b = \{\Phi_i \circ refine(b, \Phi_i) \mid \forall \Phi_i \in \mathcal{G}\}$ with \circ a revision operator. That is, revising a global uncertain belief set is taken as revising the local epistemic states with the given input. Each input (m', μ) in the sequence $refine(b, \Phi_i)$ corresponds to a simple epistemic state from [Ma and Liu, 2011], i.e. to an epistemic state Φ_{in} with the domain 2^{A_i} such that $\Phi_{in}(\omega) = \mu$ iff $\omega \models m'$ and $\Phi_{in}(\omega) = 0$ otherwise. An epistemic state Φ can be revised by a simple epistemic state Φ' with the same domain Ω , denoted as $\Phi \circ \Phi'$, as $\forall \omega \in \Omega, (\Phi \circ \Phi')(\omega) = \Phi(\omega) + \Phi'(\omega)$.⁴ As such, when the input has been transformed to $refine(b, \Phi_i)$ for a given local epistemic state Φ_i , the revision is equivalent to iterated revision using the simple epistemic states in $refine(b, \Phi_i)$. The final output of this iterated revision is unique regardless of the order in which we revise Φ_i with simple epistemic states Φ_{in} in $forget(b, \Phi_i)$ based on postulates B5 and B6 in [Ma and Liu, 2011] (i.e. weights are cumulative and the order of updating does not affect the result).

Now we can introduce the $\circ b$ rule to CAN+ for belief change. The intuition of this new rule is clear; we want to change the beliefs encoded in the GUB with the uncertain belief b . We have:

⁴For other epistemic states these values can be extrapolated.

$$\frac{}{\langle \mathcal{G}, \mathcal{A}, \circ b \rangle \longrightarrow \langle \mathcal{G} \circ b, \mathcal{A}, nil \rangle} \circ b$$

The rule for belief change can serve as a template to define the rules for classical belief addition $+\phi$ and deletion $-\phi$. Those rules would become:

$$\frac{}{\langle \mathcal{G}, \mathcal{A}, +\phi \rangle \longrightarrow \langle \mathcal{G} \circ (\phi, \max_{\mathcal{G}}), \mathcal{A}, nil \rangle} +\phi$$

$$\frac{}{\langle \mathcal{G}, \mathcal{A}, -\phi \rangle \longrightarrow \langle \mathcal{G} \circ (\phi, \min_{\mathcal{G}}), \mathcal{A}, nil \rangle} -\phi$$

with $\max_{\mathcal{G}} = \max \{\max_{\Phi_i} \mid \Phi_i \in \mathcal{G}\}$ and $\min_{\mathcal{G}}$ analogously defined. Notice that we transform the formula ϕ into an uncertain belief by assigning to it the weight $\max_{\mathcal{G}}$ ($\min_{\mathcal{G}}$). This ensures that ϕ will be true (false) after revision. We can also define belief addition and deletion as syntactic sugar on top of the belief change semantics. Indeed, a statement such as $+\phi$ is nothing more than a shorthand for the statement $\circ(\phi, \max_{\mathcal{G}})$. Similarly, $-\phi$ can be considered a shorthand for $\circ(\phi, \min_{\mathcal{G}})$. As we try to keep the semantics as concise as possible, we opt to define these operators in the latter way. Such a choice will also need to be made in the next section, where we will directly present the approach based on syntactic sugar.

In conclusion, the new language for a plan body in CAN+ is given in BNF as:

$$P ::= nil \mid \circ b \mid act \mid ?\phi \mid !e \mid P_1; P_2 \mid P_1 \parallel P_2 \mid P_1 \triangleright P_2 \mid (|\Delta|) \mid Goal(\phi_s, P, \phi_f)$$

with b an uncertain belief and $\phi, \phi_s, \phi_f \in \mathcal{L}_{\geq}$. We also modified the rules for $?\phi$ and *select*, while dropping the rules for $+\phi$ and $-\phi$ and introducing a new rule for $\circ b$. The rules in CAN dealing with program flow do not require any changes and can be integrally applied to the CAN+ semantics. The rules on declarative goals do need to be modified, but in a straightforward way similar to $?\phi$, i.e. we need to verify ϕ_s and ϕ_f against \mathcal{G} .

5 DEALING WITH UNCERTAIN ACTIONS IN A BDI AGENT

The primitive actions of a BDI agent are affected by uncertainty in a variety of ways. Usually described in a STRIPS-like style such as $act : \psi \leftarrow \phi^-; \phi^+$, an action *act* can have uncertainty in the precondition ψ , uncertainty as to a specific effect (where the effect will change the epistemic state and possibly the belief set) or uncertainty as to the outcome of an action (with a probability for each outcome).

The first form of uncertainty is the easiest to incorporate. Similar to how the rule *select* for plan selection allows us to consider uncertain information, we can take $\psi \in \mathcal{L}_{\geq}$ and verify whether this context, or precondition, is satisfied.

Next, ϕ^- and ϕ^+ are usually taken to be *delete* and *add* lists of atoms. Nothing in the semantics for CAN+ prevents

us from instead considering a list of uncertain beliefs ϕ^u as the results of an action. Not only does this considerably increase the expressive powers of action effects, but it also allows to define ϕ^- and ϕ^+ as special cases of ϕ^u with each being a list of propositions $\phi \in \mathcal{L}$ to which the weight \min_Φ and \max_Φ is assigned, respectively. As we did before, we assume hereafter that ϕ^- and ϕ^+ are forms of syntactic sugar for which we will not explicitly define the semantics.

Finally, the effects of an action may not be known in advance. This form of uncertainty has already been extensively considered in the literature, leading to variations of the STRIPS language that consider various outcomes with associated probabilities. Rather than a single outcome $\phi^-; \phi^+$ we consider a set of outcomes $\{\langle p_1, \phi_1^-, \phi_1^+ \rangle, \dots, \langle p_n, \phi_n^-, \phi_n^+ \rangle\}$ with $\sum_{i=1}^n p_i = 1$.

By adopting a STRIPS-like probabilistic action library \mathcal{D} , populated by *probabilistic action description rules* – each representing a single independent action – we can model these three forms of uncertainty with rules of the form:

$$act : \psi_{act} \leftarrow \{\langle p_1, \phi_1^u \rangle, \dots, \langle p_n, \phi_n^u \rangle\}$$

such that $p_i \geq 0$, $\sum_{i=1}^n p_i = 1$ with ψ_{act} an uncertain belief and ϕ_i^u a list of uncertain beliefs.

Example 4. Consider the running example from the introduction and Figure 1. We can model the actions to slow down and to continue at the same speed as

$$\begin{aligned} slow : true &\leftarrow \{\langle 0.4, [(junc, \max_G), (sp, -20)] \rangle, \\ &\quad \langle 0.6, [(late, \max_G), (sp, -20)] \rangle\} \\ cont : sd \geq \neg sd &\leftarrow \{\langle 0.75, [(junc, \max_G)] \rangle, \\ &\quad \langle 0.25, [(late, \max_G)] \rangle\} \end{aligned}$$

The first action can always be applied and has two outcomes. With 40% chance the junction is reached in time and with 60% the train is late. In both cases the (speed) is reduced. The second action encodes an agent in a hurry: the agent will not wait until there is a safe distance, i.e. the agent continues whenever he thinks it is at least more plausible that there is still a safe distance (or when the agent is ignorant and doesn't care).

While we already know how to correctly deal with uncertain beliefs, we do not yet have the machinery in the operational semantics to deal with probabilistic effects. To model a probabilistic action we use the notion of a probabilistic transition $C \xrightarrow{p} C'$ where p represents the transition probability between the configurations C and C' [Di Pierro and Wiklicky, 1998]. Notice that all the transition rules used thus far are special cases of probabilistic transition rules where the probability of the transition is 1. As such we assume in CAN+ that all transition rules are probabilistic transition rules, where the probability is 1 unless explicitly specified. The *act* derivation rule can then be defined as:

$$\frac{(a : \psi \leftarrow effects) \in \mathcal{D} \quad a\theta = act \quad \mathcal{G} \models \psi\theta}{\langle \mathcal{G}, \mathcal{A}, act \rangle \xrightarrow{p_i} \langle \mathcal{G} \circ \phi_i^u \theta, \mathcal{A} \cdot act, nil \rangle} act$$

with *effects* the set $\{\langle p_1, \phi_1^u \rangle, \dots, \langle p_n, \phi_n^u \rangle\}$. As expected, the transition will depend on the probabilities of the different effects associated with the action *act*.

Thus far we have only discussed how CAN+, which extends CAN, adds the ability to model and reason about uncertain information. A parallel endeavour is to extend CANPLAN into CANPLAN+. The main difference between CAN and CANPLAN is the ability of the latter to perform lookahead planning by means of the *Plan*(\cdot) action. A similar idea can be incorporated in CAN+ to arrive at CANPLAN+.

We know from the way we extended the *act* rule that, during the BDI execution, it is the probability of the transition that determines the effect of a primitive action. Furthermore, when a BDI agent tries to achieve some intention, this may involve the execution of a large number of plans. However, merely selecting the plan with the highest probability of reaching the next state without taking future actions into account may lead to poor performance. Indeed, this single step may not be on the same path that offers the highest overall chance of achieving our goal. Such issues can be addressed by using lookahead planning. During planning performed through the *Plan*(\cdot) action we can take the probability of the different transitions into account and thus maximise the probability of achieving our intention.

To formalise this idea, we introduce the notion of maximising the overall transition probability. Intuitively, given two configurations C and C'' , there may be more than one option such that $C \xrightarrow{\text{plan}^*} C''$. When a transition is labelled with ‘plan’ or ‘bdi’, the transition is resp. only valid in the planning context or during BDI execution. We want to take the transition $C \xrightarrow{\text{plan}} C'$ such that C' is the next configuration on the path which offers us the highest overall chance of reaching our goal, which we will denote as $C \xrightarrow{\text{max}^*} C''$.

Definition 9. Let C and C'' be configurations such that $C \xrightarrow{\text{plan}^*} C''$. Furthermore, let p be such that $p = \prod_{i=0}^n p_i$ and $C \xrightarrow{\text{plan}}_{p_1} C' \xrightarrow{\text{plan}}_{p_2} \dots \xrightarrow{\text{plan}}_{p_n} C''$. We say that $C \xrightarrow{\text{max}^*} C''$ when there does not exist a configuration D' that is different from C' in either its belief base, executed actions or plan body such that $C \xrightarrow{\text{plan}}_{p'_1} D' \xrightarrow{\text{plan}}_{p'_2} \dots \xrightarrow{\text{plan}}_{p'_m} C''$ and $p' > p$ with $p' = \prod_{i=0}^m p'_i$.

In other words: C' is the next configuration on the most probable path to reach C'' . Using Definition 9 we can extend the operational semantics of the *Plan*(\cdot) construct to take into account that we are dealing with uncertain actions. In CANPLAN the rule for *Plan*(\cdot) is as follows:

$$\frac{C \xrightarrow{\text{plan}} C' \quad C' \xrightarrow{\text{plan}^*} C''}{\langle \mathcal{B}, \mathcal{A}, Plan(P) \rangle \xrightarrow{\text{bdi}} \langle \mathcal{B}', \mathcal{A}', Plan(P') \rangle} plan$$

with the configurations C , C' and C'' defined as $\langle B, A, P \rangle$, $\langle B', A', P' \rangle$ and $\langle B'', A'', nil \rangle$, respectively. Intuitively, this rule states that the next action to execute is the one that, according to our lookahead planning, will *eventually* lead us to achieving our goal.

The rule in CANPLAN+ for $Plan(\cdot)$, which not only ensures that we reach our goal but also maximises the chances of reaching our goal, is then defined as:

$$\frac{C \xrightarrow{plan} C' \quad C \xrightarrow{\max^*_{C'}} C''}{\langle \mathcal{G}, A, Plan(P) \rangle \xrightarrow{bdi} \langle \mathcal{G}', A', Plan(P') \rangle} \text{ plan}$$

with C , C' and C'' defined as $\langle \mathcal{G}, A, P \rangle$, $\langle \mathcal{G}', A', P' \rangle$ and $\langle \mathcal{G}'', A'', nil \rangle$, respectively.

Example 5. Consider the running example from Figure 1. Assume we are at the decision point just after reaching the signal. An agent that plans ahead for the goal of reaching the station in time, will make the rational choice to slow down. If the agent did not perform lookahead planning and only looked at the highest chance to reach the junction, then continuing at the same speed would be preferred.

6 RELATED WORK

The BDI framework [Rao and Georgeff, 1991] is notable for treating beliefs and intentions as two distinct ideas in an agent-based setting. However, due to the complex temporal modal logic being used and the assumption of unlimited resources there was a disconnect between the theory and implementations based on BDI. This problem was mostly resolved in [Rao, 1996] where an abstract agent-based language, called AgentSpeak, was proposed. This language was strongly related to the original BDI theory, while being easily implementable.

CAN [Winikoff et al., 2002] follows up on this approach of AgentSpeak and provides operational semantics for dealing with declarative goals. Such goals allow more flexibility, e.g. plans can be stopped when the goal is reached instead of being blindly executed until the end. Declarative goals also make it easier to define semantics for planning in a BDI setting. Most approaches on planning [de Silva and Padgham, 2005, Walczak et al., 2006, Meneguzzi et al., 2007] had been ad-hoc approaches without a semantical background for the integration of planning in BDI. Such a semantical background was provided in [Sardiña et al., 2006, Sardiña and Padgham, 2011] with the introduction of CANPLAN, an extension of CAN with a $Plan(\cdot)$ action that allows for offline lookahead planning.

Notable work on the integration of uncertainty in a BDI context has been done in the setting of graded BDI [Casali et al., 2005]. In graded BDI it is assumed that the beliefs, desires and intentions have a degree of uncertainty. While of theoretical interest, their framework uses a

complex modal logic axiomatisation which makes it hard to implement the work directly. Later, in [Criado et al., 2014], the graded BDI system was further extended to incorporate norms, i.e. patterns of behaviour that should be adhered to in given circumstances. These norms are acquired and enforced in an uncertain environment. To accommodate this, norms have an associated salience to reflect their importance in the given uncertain environment.

Implementations that deal with uncertain percepts in a BDI setting [Chen et al., 2013] have not been based on the graded BDI framework but approached the problem more pragmatically. Our work extends upon the ideas of graded beliefs (i.e. what the agent knows), where we allow more fine-grained control by dividing the beliefs into isolated parts, each with their own representation and revision strategies. Contrary to graded BDI, our work has a vested interest in the feasibility of implementations while still providing strong theoretical underpinnings. In that sense, our work is close to the spirit of CANPLAN.

7 CONCLUSIONS

In this paper we showed how operational semantics for a BDI agent can be devised to deal with uncertain beliefs and actions affected by various forms of uncertainty. We introduced CANPLAN+, an extension of CANPLAN, in which we introduce a novel way of representing the agent's beliefs as a set of epistemic states. We furthermore introduced the idea of stratifying the domains of epistemic states. This allows an agent to reason about the plausibility of his beliefs within a local epistemic state and allows commensurability over the evaluation of these local results. As such, an agent can select more appropriate plans and can revise his current beliefs with uncertain information from the environment. In addition, it gives a BDI agent system designer the freedom to choose the best representation for the beliefs at hand. We also established a way to model actions triggered by uncertain beliefs, have uncertain effects and have effects that may introduce extra uncertainty into the beliefs. Finally, we extended the $Plan(\cdot)$ action from CANPLAN to allow a BDI agent to plan for the most optimal plan, i.e. with the highest chance of achieving the goal.

For future work, we plan to develop complete algorithms as well as approximate/tractable algorithms to use in BDI implementations that model and allow to reason about the uncertain beliefs of an agent. Moreover, we want to explore how existing planners under uncertainty can be extended to deal with the various forms of uncertainty faced in a SCADA environment.

Acknowledgements

This research is supported by the UK EPSRC project EP/J012149/1.

References

- [Boyer, 2009] Boyer, S. (2009). *SCADA: Supervisory Control And Data Acquisition*. International Society of Automation.
- [Bratman, 1987] Bratman, M. (1987). *Intention, Plans, and Practical Reason*. Harvard University Press.
- [Casali et al., 2005] Casali, A., Godo, L., and Sierra, C. (2005). Graded BDI models for agent architectures. In *Proc. of CLIMA'04*, pages 126–143.
- [Casali et al., 2011] Casali, A., Godo, L., and Sierra, C. (2011). A graded BDI agent model to represent and reason about preferences. *Artificial Intelligence*, 175(7–8):1468–1478.
- [Chen et al., 2013] Chen, Y., Hong, J., Liu, W., Godo, L., Sierra, C., and Loughlin, M. (2013). Incorporating PGMs into a BDI architecture. In *Proc. of PRIMA'13*, pages 54–69.
- [Criado et al., 2014] Criado, N., Argente, E., Noriega, P., and Botti, V. (2014). Reasoning about norms under uncertainty in dynamic environments. *International Journal of Approximate Reasoning*. In press.
- [Darwiche and Goldszmidt, 1994] Darwiche, A. and Goldszmidt, M. (1994). On the relation between kappa calculus and probabilistic reasoning. In *Proc. of UAI'94*, pages 145–153.
- [Dastani, 2008] Dastani, M. (2008). 2APL: a practical agent programming language. *Autonomous Agents and Multi-Agent Systems*, 16(3):214–248.
- [de Silva and Padgham, 2005] de Silva, L. and Padgham, L. (2005). Planning on demand in BDI systems. In *Proc. of ICAPS'05*, pages 37–40.
- [Di Pierro and Wiklicky, 1998] Di Pierro, A. and Wiklicky, H. (1998). An operational semantics for probabilistic concurrent constraint programming. In *Proc. of ICCL'98*, pages 174–183.
- [Dubois et al., 1994] Dubois, D., Lang, J., and Prade, H. (1994). Possibilistic logic. *Handbook of Logic for Artificial Intelligence and Logic Programming*, 3(1):439–513.
- [Dubois and Prade, 1995] Dubois, D. and Prade, H. (1995). Possibility theory as a basis for qualitative decision theory. In *Proc. of IJCAI'95*, pages 1924–1930.
- [Ingrand et al., 1992] Ingrand, F., Georgeff, M., and Rao, A. (1992). An architecture for real-time reasoning and system control. *IEEE Expert: Intelligent Systems and Their Applications*, 7(6):34–44.
- [Ma and Liu, 2011] Ma, J. and Liu, W. (2011). A framework for managing uncertain inputs: An axiomization of rewarding. *International Journal of Approximate Reasoning*, 52(7):917–934.
- [McArthur et al., 2007] McArthur, S., Davidson, E., Catterson, V., Dimeas, A., Hatziargyriou, N., Ponci, F., and Funabashi, T. (2007). Multi-agent systems for power engineering applications – part i: Concepts, approaches, and technical challenges. *IEEE Transactions on Power Systems*, 22(4):1743–1752.
- [Meneguzzi et al., 2007] Meneguzzi, F. R., Zorzo, A. F., da Costa Móra, M., and Luck, M. (2007). Incorporating planning into BDI systems. *Scalable Computing: Practice and Experience*, 8(1).
- [Rao, 1996] Rao, A. (1996). Agentspeak(1): BDI agents speak out in a logical computable language. In *Proc. of MAAMAW'96*, pages 42–55.
- [Rao and Georgeff, 1991] Rao, A. and Georgeff, M. (1991). Modeling rational agents within a BDI architecture. In *Proc. of KR'91*, pages 473–484.
- [Sardiña et al., 2006] Sardiña, S., de Silva, L., and Padgham, L. (2006). Hierarchical planning in BDI agent programming languages: a formal approach. In *Proc. of AAMAS'06*, pages 1001–1008.
- [Sardiña and Padgham, 2011] Sardiña, S. and Padgham, L. (2011). A BDI agent programming language with failure handling, declarative goals, and planning. *Autonomous Agents and Multi-Agent Systems*, 23(1):18–70.
- [Spohn, 1988] Spohn, W. (1988). Ordinal conditional functions: A dynamic theory of epistemic states. In *Causation, Coherence, and Concepts: Proceedings of Proceedings of the Irvine Conference on Probability and Causation*, pages 105–134.
- [Walczak et al., 2006] Walczak, A., Braubach, L., Pokahr, A., and Lamersdorf, W. (2006). Augmenting BDI agents with deliberative planning techniques. In *Proc. of ProMAS'06*, pages 113–127.
- [Winikoff et al., 2002] Winikoff, M., Padgham, L., Harland, J., and Thangarajah, J. (2002). Declarative & procedural goals in intelligent agent systems. In *Proc. of KR'02*, pages 470–481.
- [Zhi et al., 2000] Zhi, L., Qin, J. S., Yu, T. B., Hu, Z. J., and Hao, Z. (2000). The study and realization of scada system in manufacturing enterprises. In *Proc. of WCICA'00*, volume 5, pages 3688–3692.

Message Passing for Soft Constraint Dual Decomposition

David Belanger
UMass Amherst
belanger@cs.umass.edu

Alexandre Passos
UMass Amherst
apassos@cs.umass.edu

Sebastian Riedel
University College London
s.riedel@ucl.ac.uk

Andrew McCallum
UMass Amherst
mccallum@cs.umass.edu

Abstract

Dual decomposition provides the opportunity to build complex, yet tractable, structured prediction models using linear constraints to link together submodels that have available MAP inference routines. However, since some constraints might not hold on every single example, such models can often be improved by relaxing the requirement that these constraints always hold, and instead replacing them with soft constraints that merely impose a penalty if violated. A dual objective for the resulting MAP inference problem differs from the hard constraint problem's associated dual decomposition objective only in that the dual variables are subject to box constraints. This paper introduces a novel primal-dual block coordinate descent algorithm for minimizing this general family of box-constrained objectives. Through experiments on two natural language corpus-wide inference tasks, we demonstrate the advantages of our approach over the current alternative, based on copying variables, adding auxiliary submodels and using traditional dual decomposition. Our algorithm performs inference in the same model as was previously published for these tasks, and thus is capable of achieving the same accuracy, but provides a 2-10x speedup over the current state of the art.

1 INTRODUCTION

We often need complex structured prediction models that encode rich global and local dependencies and constraints among the outputs, but this can render efficient prediction difficult. Therefore, *dual decomposition* is quite useful, since it enables efficient inference in models composed of various submodels with available black-box MAP inference routines (Komodakis *et al.*, 2007; Sontag *et al.*, 2011; Rush & Collins, 2012).

In some cases, the flexibility and robustness of such models can be improved by using *soft constraints*, where the model imposes a cost if a constraint is violated, but does not require that it is satisfied. In natural language processing, for example, soft constraints have enabled accuracy gains for named entity recognition (Finkel *et al.*, 2005; Sutton & McCallum, 2006), parsing (Smith & Eisner, 2008; Rush *et al.*, 2012), and citation field segmentation (Chang *et al.*, 2012; Anzaroot *et al.*, 2014). Using soft constraints is reasonable in these applications because the constraints are not required in order to define feasible outputs, but are instead a modeling layer imposed to improve predictive accuracy. Soft constraints are advantageous over hard constraints because they allow the model to trade off evidence for and against a constraint being satisfied.

In all of these examples besides Rush *et al.* (2012) and Anzaroot *et al.* (2014), inference is performed using standard techniques for inference in loopy graphical models such as belief propagation or MCMC. However, these have poor optimality guarantees and can also be difficult to generalize to prediction problems that are not graphical models. An alternative method for handling soft constraints is to make copies of variables participating in soft constraints, constrain each variable to equal its copy, and apply dual decomposition (Rush *et al.*, 2012). While this exhibits better flexibility, scalability, and guarantees, it requires inference in auxiliary submodels and copying variables prevents the feasibility of the output during intermediate iterations before convergence, since the two copies of a variable may have different values.

Recently, Anzaroot *et al.* (2014) employed an attractive alternative algorithm for performing MAP subject to soft constraints that offers the optimality guarantees and generality of dual decomposition, but avoids variable copying and auxiliary models completely. Their algorithm requires an extremely straightforward modification to existing dual decomposition objectives: if the model penalizes the violation of a constraint with a penalty of c , then the dual variable is subject to a *box constraint*, where it can not exceed c . They minimize this objective with projected subgradient

descent.

While this projected subgradient algorithm is simple, its convergence can be slow and sensitive to a choice of step size schedule. On the other hand, block coordinate descent algorithms, such as MPLP (Globerson & Jaakkola, 2007), are parameter-free and often converge much faster than subgradient descent for dual decomposition objectives, subject to our ability to obtain *max-marginals* from the subproblems (Sontag *et al.*, 2011).

In response, we contribute the following:

1. An extension of the projected subgradient algorithm of Anzaroot *et al.* (2014) to general pairwise soft constraints (Section 5) that are capable of modeling arbitrary pairwise graphical model factors (Section 8).
2. An adaptation of the MPLP algorithm beyond graphical models to alternative structured prediction problems with certain structure (Section 6).
3. Box-MPLP, a primal-dual message passing algorithm for solving the box-constrained dual decomposition objective for soft constraints (Section 7). Its update rule and derivation differ substantially from MPLP.
4. Experiments on two corpus-wide prediction tasks from natural language processing (Section 2) demonstrating both the advantages of using Box-MPLP v.s. projected subgradient and of using a box-constrained dual objective v.s. variable copying and hard-constraint dual decomposition (Section 10).

2 CORPUS-WIDE INFERENCE

We first motivate the use of soft constraints by describing the application that we will explore in our experiments. In natural language processing, it is common to part-of-speech (POS) tag and dependency parse every sentence in a corpus of documents. Both tasks can be posed as efficient MAP inference, but a drawback of these algorithms is that they process each sentence in isolation, despite the fact that there is discriminative information shared across the corpus. In response, Rush *et al.* (2012) performed *corpus-wide inference*. Specifically, for word types that did not appear in the training data, they introduced global model terms that encouraged every occurrence of the word in the test corpus to receive the same POS tag, or to be assigned a dependency parent with the same POS tag. A similar model appeared in Chieu & Teow (2012).

Rush *et al.* (2012) model these cross-sentence relationships among sets of occurrences that are encouraged to agree, by introducing one *consensus structure*, described in the Figure 1 caption, per set. There is a soft constraint between every variable at the bottom of the consensus set, and the one at the top. If the underlying sentence-level models are graphical models, the corpus-wide inference problem could be posed as a large loopy graphical model and we can per-

Figure 1: One *consensus set*. The circles at the bottom represent words of the same type, and the boxes represent arbitrary sentence-level prediction problems that they are contained in. The circle at the top is a *consensus variable* introduced to encourage consensus among the bottom circles, where the squares are soft constraints penalizing disagreement. The corpus is linked together by a web of consensus structures.

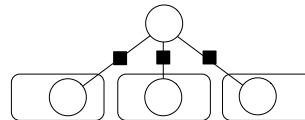
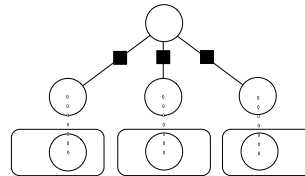


Figure 2: The variable-copying version of Fig. 1, where dashed lines denote equality constraints.



form approximate MAP using standard techniques. An alternative solution, depicted in Figure 2, is to copy variables that participate in consensus sets, introduce an auxiliary tree-structured subproblem, and use dual decomposition for corpus-wide MAP. This has superior optimality guarantees and flexibility to use sentence-level problems that are not graphical models. In practice, this algorithm can be slow to converge, however. In response, we introduce a new approach for performing MAP subject to soft constraints that when applied to corpus-wide inference allows us to work directly in the soft constraint problem of Figure 1, yet yields the same flexibility and optimality guarantees as Rush *et al.* (2012) and substantially faster runtimes. The techniques are general and apply to a wide range of additional applications.

3 NOTATION AND STRUCTURED LINEAR MODELS

Bold-faced lower-case letters, such as \mathbf{x} , represent column vectors, and bold-faced upper case letters, such as \mathbf{A} , represent matrices. The i -th coordinate of vector \mathbf{x} is $\mathbf{x}(i)$ and the i, j th coordinate of a matrix is $\mathbf{A}(i, j)$. Lower-case greek letters such as λ represent either vector-valued or matrix-valued dual variables. We use $\mathbf{x}^{(t)}$ for \mathbf{x} at iteration t . The term 'constraint' either refers to a constraint between scalars or a set of coordinate-wise constraints between vectors (or matrices). In the latter case, the associated dual variable is a vector (or matrix).

We consider structured prediction problems defined by

structured linear models such as conditional random fields (Lafferty *et al.*, 2001) and maximum spanning tree parsers (McDonald *et al.*, 2005). These assign a score to each possible output labeling by decomposing each candidate output into a collection of *parts*, each of which can be active or inactive in a given labeling. For example, in first-order dependency parsing, each part corresponds to a single dependency arc (Smith, 2011). In a conditional random field, there is a part for each possible setting of each clique.

We write the indicator vector for the parts of a specific labeling of a dataset k as \mathbf{x}_k . It is a binary vector with one coordinate per possible part, which is zero if the part is not present in the structured output and one if it is. The model for candidate outputs is called *linear* because the score of a given labeling is the dot product $\langle \mathbf{w}_k, \mathbf{x}_k \rangle$ of a weight vector \mathbf{w}_k and the indicator vector over the parts. In many models, such as conditional random fields, the score of each part is a function of some observed features, and in many cases this mapping from features to weights is also linear. We focus only on inference, however, and make no assumptions about how the weights are set. In non-trivial structured linear models, not all assignments of values to these parts are valid, since they typically represent some over-complete view of the structured output or are subject to global structural constraints, such as projectivity for dependency parsing (Smith, 2011). For an instance k we refer to the set of valid assignments to parts as \mathcal{U}_k .

We refer to the problem of finding the highest-scoring valid collection of parts as *MAP inference*:

$$\max_{\mathbf{x}_k} \langle \mathbf{w}_k, \mathbf{x}_k \rangle \quad \text{s.t.} \quad \mathbf{x}_k \in \mathcal{U}_k.$$

4 DUAL DECOMPOSITION

Following Sontag *et al.* (2011); Rush & Collins (2012); Komodakis *et al.* (2007), we consider the problem:

$$\max_{\mathbf{x}} \sum_k \langle \mathbf{w}_k, \mathbf{x}_k \rangle \quad (1)$$

$$\text{s.t.} \quad \forall k \quad \mathbf{x}_k \in \mathcal{U}_k \quad (2)$$

$$\sum_k \mathbf{A}_k \mathbf{x}_k = 0, \quad (3)$$

where each \mathbf{x}_k represents the vector of parts for a specific structured linear ‘submodel.’ The formulation can easily be adapted to account for a nonzero right hand side of (3). If (3) did not exist, the problem would reduce to independent MAP inference in each subproblem.

Dualizing the linear constraints in (3), but not the $\mathbf{x}_k \in \mathcal{U}_k$ constraints, results in the Lagrange dual problem:

$$\min_{\boldsymbol{\lambda}} D(\boldsymbol{\lambda}) = \sum_k \max_{\mathbf{x}_k \in \mathcal{U}_k} \langle \mathbf{w}_k + \mathbf{A}_k^T \boldsymbol{\lambda}, \mathbf{x}_k \rangle. \quad (4)$$

Algorithm 1 Dual Decomposition with Subgradient Descent

```

1:  $\boldsymbol{\lambda} \leftarrow \mathbf{0}$ 
2: while has not converged do
3:   for submodel  $i$  do
4:      $\mathbf{x}_k^* \leftarrow \max_{\mathbf{x}_k \in \mathcal{U}_k} \langle \mathbf{w}_k + \mathbf{A}_k^T \boldsymbol{\lambda}, \mathbf{x}_k \rangle$ 
5:    $\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} - \eta^{(t)} \sum_k \mathbf{A}_k \mathbf{x}_k^*$ 

```

The dual objective $D(\boldsymbol{\lambda})$ is convex and piece-wise linear, as it is the sum of the supremum of linear functions of $\boldsymbol{\lambda}$, and hence can be solved with known convex optimization techniques, including subgradient methods. Any particular element of the subgradient of the dual function with respect to $\boldsymbol{\lambda}$ can be written as

$$\partial D(\boldsymbol{\lambda}) = \sum_k \mathbf{A}_k \mathbf{x}_k^*, \quad (5)$$

where each \mathbf{x}_k^* is some maximizer of a MAP inference problem with shifted weights:

$$\mathbf{x}_k^* \in \operatorname{argmax}_{\mathbf{x}_k \in \mathcal{U}_k} \langle \mathbf{w}_k + \mathbf{A}_k^T \boldsymbol{\lambda}, \mathbf{x}_k \rangle. \quad (6)$$

We consider cases, where these MAP subproblems are tractable and solving their linear programming relaxations returns an integral value for any weight vector. Therefore, one can use subgradient descent, Algorithm 1, to minimize the dual problem. Subject to conditions on the sequence of step sizes $\eta^{(t)}$ and the feasibility of the constraints that link the subproblems, the subgradient method is guaranteed to converge to the optimum, where (3) will be satisfied (Nesterov, 2003; Sontag *et al.*, 2011).

5 SOFT DUAL DECOMPOSITION

5.1 PROBLEM STATEMENT

This paper focuses on applications of dual decomposition where the underlying prediction problem has at least two distinct sets of outputs $\mathbf{x}_1 \in \mathcal{U}_1$ and $\mathbf{x}_2 \in \mathcal{U}_2$, and linear constraints are imposed between them not as a requirement to define feasible outputs, but as an extra layer of modeling to encourage global regularity of the outputs. This contrasts with problems with a single output \mathbf{x} subject to the linear constraints $\mathbf{x} \in \mathcal{U}_1 \cap \mathcal{U}_2$, and while these are unmanageable directly, \mathcal{U}_1 and \mathcal{U}_2 can each be handled in isolation. Here, dual decomposition can be employed via a copy variable \mathbf{x}_2 , and constraints $\mathbf{x} \in \mathcal{U}_1$, $\mathbf{x}_2 \in \mathcal{U}_2$, and $\mathbf{x}_1 = \mathbf{x}_2$ (Koo *et al.*, 2010; Rush & Collins, 2012). The first family is precisely where it can make sense to employ soft constraints, since they will not threaten the output’s feasibility.

Anzaroot *et al.* (2014) recently performed MAP with soft constraints by performing projected gradient descent in a box-constrained dual objective. Our message passing algorithm requires using a slightly more restrictive set of global

constraint structures to be converted into soft constraints than what they considered, which are of the form (3). Specifically, we assume the global constraints decompose into sets of pairwise equality constraints between components of submodels:

$$\max_{\mathbf{x}} \quad \sum_k \langle \mathbf{w}_k, \mathbf{x}_k \rangle \quad (7)$$

$$\text{s.t.} \quad \forall k \quad \mathbf{x}_k \in \mathcal{U}_k \quad (8)$$

$$\forall (\mathbf{A}_p, \mathbf{B}_p, p_1, p_2) \in \mathcal{P} \quad \mathbf{A}_p \mathbf{x}_{p_1} = \mathbf{B}_p \mathbf{x}_{p_2}. \quad (9)$$

A given product $\mathbf{A}_p \mathbf{x}_{p_1}$ or $\mathbf{B}_p \mathbf{x}_{p_2}$ is allowed to appear in multiple $p \in \mathcal{P}$, so \mathcal{P} effectively defines a collection of linear measurements of the structured output and a graph of equality constraints among them. These can be defined over differently-size mapping matrices. Define s_p to be the length of the vector $\mathbf{A}_p \mathbf{x}_{p_1}$ (also the length of $\mathbf{B}_p \mathbf{x}_{p_2}$).

Defining a dual variable $\lambda_p \in \mathbb{R}^{s_p}$ for every $p \in \mathcal{P}$, we have the following convex dual decomposition objective:

$$\sum_k \max_{\mathbf{x}_k} \left\langle \mathbf{w}_k + \sum_{p:p_1=k} \mathbf{A}_p^T \lambda_p - \sum_{p:p_2=k} \mathbf{B}_p^T \lambda_p, \mathbf{x}_k \right\rangle. \quad (10)$$

A soft constraint formulation of (7) with penalty matrices $\mathbf{c}_p \in \mathbb{R}^{s_p \times s_p}$ subtracts a penalty of $\mathbf{c}_p(i, j)$ from the score of the global MAP problem whenever $\mathbf{A}_p \mathbf{x}_{p_1}$ is set to value i and $\mathbf{B}_p \mathbf{x}_{p_2}$ is *not* set to value j . In the subsequent exposition, we leave the constraints $\mathbf{x}_k \in \mathcal{U}_k$ implicit, since we assume we have available black-box algorithms for maximizing over these constraint sets. Therefore, we have:

$$\max_{\mathbf{x}} \sum_k \langle \mathbf{w}_k, \mathbf{x}_k \rangle - \sum_p \sum_{i,j} \mathbf{c}_p(i, j) [\mathbf{A}_p \mathbf{x}_{p_1}(i) - \mathbf{B}_p \mathbf{x}_{p_2}(j)]_+ \quad (11)$$

where $[\cdot]_+ = \max(0, \cdot)$. Using a matrix-valued penalty is important in order to support a mapping between arbitrary graphical model factors and soft constraints (see Section 8). In Section 7.1, we consider diagonal \mathbf{c}_p , which are sufficient for the model to penalize when certain components of the structured output do not take on the same value.

An alternative to (11) for expressing soft constraints is to create copies of both of the terms appearing in each $p \in \mathcal{P}$ and enforce the constraints that terms equal their copy:

$$\begin{aligned} \max_{\mathbf{x}} \quad & \sum_k \langle \mathbf{w}_k, \mathbf{x}_k \rangle - \sum_p \sum_{i,j} \mathbf{c}_p(i, j) [\mathbf{v}_p(i) - \mathbf{u}_p(j)]_+ \\ \text{s.t.} \quad & \forall p \in \mathcal{P} \quad \mathbf{A}_p \mathbf{x}_{p_1} = \mathbf{v}_p, \mathbf{B}_p \mathbf{x}_{p_2} = \mathbf{u}_p. \end{aligned} \quad (12)$$

Here, the second term is not a structured linear model, but it is concave, can be handled efficiently in isolation, and

has integral optima. Therefore, we can apply standard dual decomposition techniques. In Figure 2, we demonstrate how Rush *et al.* (2012) similarly use variable copying to make MAP tractable with dual decomposition. Rather than employing pairwise hinge losses as auxiliary submodels, they introduce a single tree-structured graphical model with pairwise factors that encourage agreement. In Section (10) we use this as a baseline to demonstrate the deficiencies of using variable copying to implement soft constraints.

5.2 DUAL OBJECTIVE AND BOX CONSTRAINTS

Problem (11) can be rewritten as a linear program by introducing matrices of auxiliary variables $\mathbf{z}_p \in \mathbb{R}^{s_p \times s_p}$:

$$\max_{\mathbf{x}, \mathbf{z}} \quad \sum_k \langle \mathbf{w}_k, \mathbf{x}_k \rangle - \sum_p \sum_{i,j} \mathbf{c}_p(i, j) \mathbf{z}_p(i, j) \quad (13)$$

$$\begin{aligned} \text{s.t.} \quad & \forall (i, j), \quad \mathbf{z}_p(i, j) \geq \mathbf{A}_p \mathbf{x}_{p_1}(i) - \mathbf{B}_p \mathbf{x}_{p_2}(j) \\ & \mathbf{z}_p \geq 0 \end{aligned} \quad (14)$$

This problem is well-defined only if \mathbf{c}_p is non-negative in every coordinate. In this case, we have that problems (11) and (13) have the same optimal value and maximizing \mathbf{x} .

We defer a full derivation of the associated Lagrange dual problem for (13) to Appendix 1, since it parallels Anza-root *et al.* (2014). The dual is similar to (10), but imposes coordinate-wise box constraints:

$$\begin{aligned} \min_{\boldsymbol{\nu}} \quad & \sum_k \max_{\mathbf{x}_k} \left\langle \mathbf{w}_k + \sum_{p:p_2=k} \mathbf{B}_p^T \boldsymbol{\nu}_p^T \mathbf{1} - \sum_{p:p_1=k} \mathbf{A}_p^T \boldsymbol{\nu}_p \mathbf{1}, \mathbf{x}_k \right\rangle \\ \text{s.t.} \quad & 0 \leq \boldsymbol{\nu}_p \leq \mathbf{c}_p. \end{aligned} \quad (15)$$

Unlike for hard constraints, we have a matrix-valued dual variable $\boldsymbol{\nu}_p \in \mathbb{R}_+^{s_p \times s_p}$ for every $p \in \mathcal{P}$, where $\nu_p(i, j)$ corresponds to the constraint in (14) for a particular (i, j) , and \mathbb{R}_+ denotes the non-negative real numbers. We use $\mathbf{1}$ to be a column vector of all ones, where its length is determined by the context.

These box constraints exist for the same reason that they occur in the dual problem for soft-margin SVMs (Cortes & Vapnik, 1995), since the second term in (11) is a sum of negative hinge losses. The box constraints on the dual variables $\boldsymbol{\nu}$ can be interpreted as the Lagrangian penalizing the violation of constraints, but only so much as the primal problem would penalize their violation.

The only qualitative difference between the dual problems in (15) and (4) is the box constraints. Therefore, we can employ the projected subgradient method, shown in Algorithm 2, which will converge to the global MAP optimum if \mathcal{P} is feasible. At the end of Appendix 1, we derive the following complementary slackness criteria used for detecting convergence. These will hold for every $p \in \mathcal{P}$ and every coordinate pair (i, j) when maximizing over the primal variables:

Algorithm 2 Projected subgradient soft dual decomposition for general matrix-valued soft constraint penalties.

```

1:  $\nu \leftarrow 0$ 
2: while has not converged do
3:   for submodel  $k$  do
4:      $\tilde{\mathbf{w}}_k \leftarrow \mathbf{w}_k + \sum_{p:p_2=k} \mathbf{B}_p^T \nu_p^T \mathbf{1} - \sum_{p:p_1=k} \mathbf{A}_p^T \nu_p \mathbf{1}$ 
5:      $\mathbf{x}_k^* \leftarrow \max_{\mathbf{x}_k \in \mathcal{U}_k} \langle \tilde{\mathbf{w}}_k, \mathbf{x}_k \rangle$ 
6:   for soft constraint  $p \in \mathcal{P}$  do
7:      $\nu^p(i, j) \leftarrow \min(\mathbf{c}_p(i, j), \max(0, \nu_p(i, j) - \eta^{(t)}(\mathbf{A}_p \mathbf{x}_{p_1}^*(i) - \mathbf{B}_p \mathbf{x}_{p_2}^*(j))))$ 

```

$$\begin{aligned}
&\text{either } \mathbf{A}_p \mathbf{x}_{p_1}^*(i) = \mathbf{B}_p \mathbf{x}_{p_2}^*(j) \\
&\text{or } \mathbf{A}_p \mathbf{x}_{p_1}^*(i) = 1 \text{ and } \nu_p(i, j) = 0 \\
&\text{or } \mathbf{A}_p \mathbf{x}_{p_1}^*(i) = 0 \text{ and } \nu_p(i, j) = \mathbf{c}_p(i, j).
\end{aligned} \tag{16}$$

6 MAX-MARGINALS AND MPLP

Using the subgradient method in Algorithm 2 is undesirable due to its sensitivity to step-size schedule and slow convergence in practice. In response, we now revisit hard-constraint dual objectives of the form (10) in order to explore previous use of block coordinate descent, which is parameter-free. We introduce an adaptation of the MPLP algorithm (Globerson & Jaakkola, 2007) to problems with general structured linear models as subproblems, and emphasize a primal-dual interpretation of the algorithm’s updates, which we will draw on when we derive our new algorithm in the following section.

MPLP is a convergent alternative to max-product belief propagation that was shown in Sontag *et al.* (2011) to be performing block coordinate descent in a dual decomposition objective for a certain instance of (10). Specifically, there is a submodel for every node and every factor in a factor graph, and an element $p \in \mathcal{P}$ between every node and every factor that it touches. MPLP generalizes to additional cases (10) when the elements of \mathcal{P} satisfy the following condition, and when the subproblems admit efficient computation of max-marginals, defined below.

Definition Let e_j denote the vector that is all zeros, except for a one in the j th coordinate. We say that the product $\mathbf{A}\mathbf{x}_k$ is a *projection variable* if it satisfies the following property:

$$\forall \mathbf{x}_k \in \mathcal{U}_k, \exists j \text{ s.t. } \mathbf{A}\mathbf{x}_k = e_j. \tag{17}$$

Unlike the previous subgradient algorithms, MPLP requires every element of \mathcal{P} to be defined between projection variables, which can be used to represent any set of mutually-exclusive states of the structured output. This is

not a strong restriction, as they can be used, for example, to zoom in on a specific graphical model node or dependency parse arc and to optionally further coarsen the values of these individual outputs. Also, the hinge loss of the previous section and 0-1 loss are equivalent for projection variables, so we are truly penalizing the event that a constraint is violated, and not imposing a linear penalty on the degree to which it is violated. Defining projection variables is necessary because MPLP requires max-marginals, and the following definition is only well-posed for projection variables:

Definition For a given projection variable $\mathbf{A}\mathbf{x}_k$ and weight vector \mathbf{w} , the max-marginals $\mathbf{m}_{\mathbf{w}}^{\mathbf{A}}$ are a vector where the j th component is given by best possible score achievable by a valid structured output when the projection variable takes on value j , i.e.,

$$\mathbf{m}_{\mathbf{w}}^{\mathbf{A}}(j) = \max_{\mathbf{x}_k \in \mathcal{U}_k} \langle \mathbf{w}, \mathbf{x}_k \rangle \text{ s.t. } \mathbf{A}\mathbf{x}_k = e_j. \tag{18}$$

For a MAP assignment \mathbf{x}^* with respect to \mathbf{w} , we have

$$\mathbf{A}\mathbf{x}^* = e_{i^*}, \text{ where } i^* = \operatorname{argmax}_i \mathbf{m}_{\mathbf{w}}^{\mathbf{A}}(i). \tag{19}$$

In other words, locally maximizing max-marginals is equivalent to finding a globally-optimal value (unless there are ties in the max-marginals).

Furthermore, max-marginals change linearly with respect to changes to \mathbf{w} in the direction of their projection variable:

$$\mathbf{m}_{\mathbf{w} + \mathbf{A}^T \alpha}^{\mathbf{A}}(i) = \mathbf{m}_{\mathbf{w}}^{\mathbf{A}}(i) + \alpha(i). \tag{20}$$

For example, if we shift the scores for a given factor in a graphical model by a vector α , and otherwise leave the model’s potentials unchanged, then the max-marginals for this factor increase by exactly α . This fact, proven in Appendix 2, applies to arbitrary projection variables, and is crucial in deriving both MPLP and our new algorithm in the next section.

In Algorithm 3, we consider a version of MPLP where block coordinate descent is performed by iteratively selecting an element $p \in \mathcal{P}$ and updating the vector-valued dual variable λ_p . Note this differs from the algorithms in Globerson & Jaakkola (2007) and Sontag *et al.* (2011) slightly because we pass messages (i.e., dual variables) directly between submodels, rather than from submodels to primal variables and from primal variables to submodels. This results from the fact that we pose (10) via equality constraints between different parts of the structured output, not between variables and their copies (Werner, 2008).

We discuss the optimality of this choice of λ_p in more detail in Appendix 3, which presents a different primal-dual argument than Sontag *et al.* (2011), in order to motivate the techniques used by the new algorithm that we will introduce later. The high level idea is to invoke (20) to observe that the chosen value for λ_p shifts the subproblems’

Algorithm 3 An adaptation of the MPLP algorithm of Sontag *et al.* (2011) to dual decomposition with pairwise constraints between general structured linear submodels.

```

1:  $\lambda \leftarrow \mathbf{0}$ 
2:  $\text{converged} \leftarrow \text{false}$ 
3: while ( $\neg \text{converged}$ ) and ( $\text{iteration} < \text{maxIterations}$ ) do
4:    $\text{converged} \leftarrow \text{true}$ 
5:   for equality constraint  $p \in \mathcal{P}$  do
6:      $\tilde{w}_{p_1} \leftarrow \mathbf{w}_{p_1} + \sum_{\substack{p': p'_1=p_1 \\ p' \neq p}} \mathbf{A}_{p'}^T \lambda_{p'} - \sum_{\substack{p': p'_2=p_1 \\ p' \neq p}} \mathbf{B}_{p'}^T \lambda_{p'}$ 
7:      $\mathbf{m}_1 \leftarrow \text{MaxMargs}(\tilde{w}_{p_1})$ 
8:      $\tilde{w}_{p_2} \leftarrow \mathbf{w}_{p_2} + \sum_{\substack{p': p'_1=p_2 \\ p' \neq p}} \mathbf{A}_{p'}^T \lambda_{p'} - \sum_{\substack{p': p'_2=p_2 \\ p' \neq p}} \mathbf{B}_{p'}^T \lambda_{p'}$ 
9:      $\mathbf{m}_2 \leftarrow \text{MaxMargs}(\tilde{w}_{p_2})$ 
10:    if ( $\text{argmax}_i \mathbf{m}_1(i) \cap \text{argmax}_i \mathbf{m}_2(i) = \emptyset$ ) then
11:       $\text{converged} \leftarrow \text{false}$ 
12:     $\lambda_p \leftarrow \frac{1}{2} (\mathbf{m}_1 - \mathbf{m}_2)$ 

```

weights such that max-marginals for the two projection variables in p become identical in all coordinates. Therefore, with this setting of the dual variables, it is feasible to achieve the equality $\mathbf{A}_p \mathbf{x}_{p_1} = \mathbf{B}_p \mathbf{x}_{p_2}$ when maximizing over the primal variables. As a result, by strong duality, the dual of (7) is minimized with respect to λ_p , since the primal constraints for this block are satisfied. Algorithm 3 monitors convergence by checking if all constraints are satisfied when maximizing over the primal variables. See Sontag *et al.* (2011) for a discussion of the convergence guarantees of MPLP and Meshi *et al.* (2012) for its convergence rate.

The algorithm may require multiple passes to converge, since updates for one λ_p may break the above optimality condition for other $p \in \mathcal{P}$. Furthermore, every time the dual variables are updated for some $p \in \mathcal{P}$, max-marginals need to be recalculated for subproblems p_1 and p_2 . MPLP, and the algorithm in the next section, can not be applied for constraints between projection variables in the same submodel, since their max-marginals interact with each other. Therefore, it could not have been applied in the hard constraint experiments of Anzaroot *et al.* (2014), since they impose constraints within a chain-structured graphical model.

7 MESSAGE PASSING FOR SOFT CONSTRAINT DUAL DECOMPOSITION

We now introduce the primary contribution of the paper: a general dual block coordinate descent framework for minimizing the box-constrained dual objective (15) and Box-MPLP, a novel algorithm for solving a common special case of the problem. Naively applying the MPLP updates may violate the box constraints, and we can not simply follow them with a projection step, as this will not guarantee a decrease in the dual objective.

Analogous to Algorithm (3), our block coordinate descent steps update one vector ν_p at a time. Since we now focus on a specific $p \in \mathcal{P}$, we define $\mathbf{y}_1 := \mathbf{A}_p \mathbf{x}_{p_1}$, $\mathbf{y}_2 := \mathbf{B}_p \mathbf{x}_{p_2}$. While MPLP is a purely dual algorithm, i.e., the update to λ_p in Algorithm 3 line 12 does not require reasoning about optimal settings of the corresponding primal variables, Box-MPLP requires explicitly constructing a primal-dual pair.

The algorithm has two overall steps (a) fixing all dual variables besides ν_p , define a small block-specific optimization problem, and efficiently determine what the optimal values \mathbf{y}_1^* and \mathbf{y}_2^* should be for it, and (b) construct a value for ν_p^* for which maximizing over the primal variables yields the values determined in step (a) and satisfies the complementary slackness conditions (16) (a). Therefore, by construction of a primal-dual certificate, ν_p^* minimizes the block coordinate descent objective.

In step (a), we seek primal optimizers \mathbf{y}_1^* and \mathbf{y}_2^* . With all dual variables besides ν_p fixed, MAP inference in the subproblems p_1 and p_2 is with respect to shifted weight vectors \tilde{w}_{p_1} and \tilde{w}_{p_2} as defined in Algorithm 3 lines 6 and 8 (which doesn't include ν_p in the shift). Using (19) we can reduce the choice of \mathbf{y}_1^* and \mathbf{y}_2^* to a local optimization problem by obtaining max-marginals \mathbf{m}_1 and \mathbf{m}_2 for the subproblems, as in Algorithm 3 lines 7 and 9. With these, we have $(\mathbf{y}_1^*, \mathbf{y}_2^*) = (e_{i^*}, e_{j^*})$, where

$$(i^*, j^*) = \arg \max_{(i,j)} \mathbf{m}_1(i) + \mathbf{m}_2(j) - \sum_{j' \neq j} \mathbf{c}_p(i, j'). \quad (21)$$

Step (b) constructs a ν_p^* that satisfies (16) and for which optimizing over the primal variables yields $(\mathbf{y}_1, \mathbf{y}_2) = (i^*, j^*)$. Invoking the ‘linearity’ of max-marginals (20), this can be expressed as the following conditions on ν_p :

$$\forall i, \mathbf{m}_1(i^*) - \sum_j \nu_p(i^*, j) \geq \mathbf{m}_1(i) - \sum_j \nu_p(i, j) \quad (22)$$

$$\forall j, \mathbf{m}_2(j^*) + \sum_i \nu_p(i, j^*) \geq \mathbf{m}_2(j) + \sum_i \nu_p(i, j). \quad (23)$$

Satisfying (16) along with (22) and (23) ensures that the independent maximizations of the reweighted problems will have the same score and same maximizing values as the joint maximization in equation (21), and thus we have a primal-dual pair for the coordinate descent subproblem.

Solving the maximization in (21) can be done, in the worst case, by enumerating all s_p^2 possible i and j . Selecting ν_p that satisfies conditions (16), (22), and (23) requires solving a linear feasibility problem, however. While this can be done in time polynomial in s_p , we focus in the next section on an important special case where it is particularly tractable, and leave exploration of general algorithms for this feasibility problem to future work.

7.1 AGREEMENT FACTORS

Next, we focus on a particular structure of \mathbf{c}_p that is both reasonable for applications and for which finding ν_p satisfying (16), (22), and (23) can be done in time $O(s_p)$. This results in the block coordinate descent Algorithm 4.

Definition Let \mathbf{y}_1 and \mathbf{y}_2 be two projection variables with values i and j , and define vector $\alpha \in \mathbb{R}_+^{s_p}$. An *agreement factor* between \mathbf{y}_1 and \mathbf{y}_2 is a structured linear model that assigns a score of 0 if they agree and a score of $-\alpha(i)$ if they disagree. This is equivalent to a penalty matrix:

$$\mathbf{c}_p(i, j) = \begin{cases} \alpha(i) & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases} \quad (24)$$

For many applications, it is sufficient to use agreement factors rather than full matrix penalties $\mathbf{c}_p(i, j)$, since they allow the model to impose a penalty if two components of the structured output are not equal. This, for example, supports the soft constraints of Rush *et al.* (2012) that we employ in our experiments. However, we show in Section 8 that matrix penalties are important to support a mapping between general graphical model factors and soft constraints.

Given the structure (24) on the penalties, there are effectively only s_p dual variables in the matrix ν_p , as the off-diagonal elements are constrained to be equal to 0 by the box constraints (15). We refer to the dual variable and costs as $\nu_p(i)$ and $\mathbf{c}_p(i)$, and equations (22) and (23) reduce to

$$\mathbf{m}_1(i^*) - \nu_p(i^*) \geq \mathbf{m}_1(i) - \nu_p(i) \quad \forall i, j \quad (25)$$

$$\mathbf{m}_2(j^*) + \nu_p(j^*) \geq \mathbf{m}_2(j) + \nu_p(j) \quad \forall i, j \quad (26)$$

In Appendix 4 we derive an $O(s_p)$ method for choosing ν_p that satisfies (16), (22), and (23). The overall insight is that (25) and (26) can be manipulated to yield simple upper and lower bounds on feasible values of $\nu_p(i)$ for $i \neq i^*, j^*$, for which we choose the midpoint of the feasible interval (Algorithm 4, line 22). Also, if $i^* \neq j^*$, then $\nu_p(i^*)$ and $\nu_p(j^*)$ are determined by complementary slackness (line 18) and otherwise, we can set them by similarly taking the mid-point of a feasible interval obtained from (25) and (26) (line 15). If we make the further restriction that the agreement factor uniformly penalizes disagreement between values of \mathbf{y}_1 and \mathbf{y}_2 , i.e. \mathbf{c}_p is α in all coordinates, then we have the added benefit that Algorithm 4 line 11 can be solved in $O(s_p)$ time. See the end of Appendix 4.

8 SOFT CONSTRAINTS V.S. FACTORS

As identified in the introduction, a traditional way to model soft constraints is to add global factors to a graphical model. In this case, the factors contribute scores when variables are set to certain values, which differs from our

Algorithm 4 Box-MPLP: block coordinate descent for soft dual decomposition with agreement factors.

```

1: converged  $\leftarrow$  false
2: while !converged do
3:   converged  $\leftarrow$  true
4:   for constraint  $p \in \mathcal{P}$  do
5:      $\tilde{w}_{p_1} \leftarrow \mathbf{w}_{p_1} + \sum_{\substack{p': p'_2=p_1 \\ p' \neq p}} \mathbf{B}_{p'}^T \nu_{p'} - \sum_{\substack{p': p_1=p_1 \\ p' \neq p}} \mathbf{A}_{p'}^T \nu_{p'}$ 
6:      $\mathbf{m}_1 \leftarrow \text{MaxMargs}(\tilde{w}_{p_1})$ 
7:      $\tilde{w}_{p_2} \leftarrow \mathbf{w}_{p_2} + \sum_{\substack{p': p'_2=p_2 \\ p' \neq p}} \mathbf{B}_{p'}^T \nu_{p'} - \sum_{\substack{p': p_1=p_2 \\ p' \neq p}} \mathbf{A}_{p'}^T \nu_{p'}$ 
8:      $\mathbf{m}_2 \leftarrow \text{MaxMargs}(\tilde{w}_{p_2})$ 
9:     if (16) not satisfied then
10:       converged  $\leftarrow$  false
11:        $i^*, j^* \leftarrow \underset{i, j}{\text{argmax}} \mathbf{m}_1(i) + \mathbf{m}_2(j) - \mathbf{c}_p(i) \delta(i \neq j)$ 
12:       if  $i^* = j^*$  then
13:          $U \leftarrow \min_{i \neq i^*} \mathbf{m}_1(i^*) - \mathbf{m}_1(i)$ 
14:          $L \leftarrow \max_{j \neq j^*} \mathbf{m}_2(j^*) - \mathbf{m}_2(j^*) + \mathbf{c}_p(j)$ 
15:          $\nu_p(i^*) \leftarrow \frac{1}{2}(U + L)$ 
16:       else
17:          $\nu_p(i^*) \leftarrow 0$ 
18:          $\nu_p(j^*) \leftarrow \mathbf{c}_p(j^*)$ 
19:       for all  $i$  such that  $i \neq i^*, i \neq j^*$  do
20:          $L \leftarrow -\mathbf{m}_1(i) + \mathbf{m}_1(i^*) + \nu_p(i^*)$ 
21:          $U \leftarrow \mathbf{m}_2(j^*) - \mathbf{m}_2(j) + \nu_p(j^*)$ 
22:          $\nu_p(i) \leftarrow \frac{1}{2}(U + L)$ 

```

use of penalties that contribute negative score when variables are *not* set to certain values. We prove in Appendix 5 that the expressivity of factors and our soft constraints are equivalent, though, as long as the soft constraints are defined between projection variables. Specifically, any table of factor scores can be mapped into a penalty matrix \mathbf{c}_p by solving an associated linear system. This may require using Algorithm 2 for inference, though, since Box-MPLP only applies to diagonal \mathbf{c}_p .

Though the two formulations are similar, soft constraints have attractive properties compared to factors. For example our algorithms maintain primal feasibility during intermediate iterations and avoid variable copying, which fractures the evidence for variables' MAP values across sub-models and requires an entire dual decomposition iteration for information to travel between output variables and their copies. Our experiments support the desirability of avoiding variable copying. In future work, we will explore solving problems that are natively expressed using factors by first mapping them to problems with soft constraints.

9 RELATED WORK

There is a precedent for constructing message passing schemes for inference problems by minimizing an associated dual problem that decomposes into local interactions (Wainwright *et al.*, 2005; Komodakis *et al.*, 2007;

Globerson & Jaakkola, 2007; Ravikumar *et al.*, 2010; Martins *et al.*, 2012; Schwing *et al.*, 2012). Many of these are based on block coordinate descent. The generalizations we make in Section 6, such as working in terms of projection variables to make MPLP apply to more general structured prediction problems than graphical models, could also be applied to a variety of these other algorithms, where the requirement that the subproblems yield max-marginals would be replaced with other requirements, such as the ability to perform MAP in the presence of additional strongly-convex terms. Our algorithm, particularly in the context of the application we consider in the next section, can also be seen as an example of special-case handling of factors that have a specific combinatorial structure (Duchi *et al.*, 2007; Martins *et al.*, 2012; Mezzumani *et al.*, 2013).

Our message passing algorithm has the same optimality guarantees as those for MPLP discussed in Sontag *et al.* (2011). Unlike (projected) subgradient descent, block coordinate descent may return sub-optimal outputs because our objective is non-smooth and not strongly convex (Luo & Tseng, 1992). Analysis of the convergence rate for smoothed versions of MPLP (Meshi *et al.*, 2012) is doable, however, and we encourage exploration of (smoothed) parallel versions of Box-MPLP (Richtárik & Takáč, 2012).

10 EXPERIMENTS

We evaluate soft constraint algorithms that vary along two dimensions: whether they solve box-constrained dual decomposition objectives or unconstrained ones based on variable copying and whether they employ (projected) subgradient descent or block coordinate descent. The first dimension is captured by the distinction between Figure 1, where the consensus variable at the top is an isolated structured linear model and there are soft constraints between this and the variables in the sentences, and Figure 2, which requires variable copying and an auxiliary tree-structured submodel. While Rush *et al.* (2012) did not employ MPLP, max-marginals can be obtained for the CRF tagger and projective parser they used (Smith, 2011). Also, note that the soft constraint penalties of Rush *et al.* (2012) used in both figures take the form of agreement factors. Therefore, we can apply Box-MPLP. We compare:

- Subgradient: Algorithm 1 applied to Figure 2
- Box-Subgradient: Algorithm 2 applied to Figure 1
- MPLP: Algorithm 3 applied to Figure 2
- Box-MPLP: Algorithm 4 applied to Figure 1

The specific problem considered by Anzaroot *et al.* (2014) problem does not admit a baseline algorithm that uses variable copying and hard-constraint dual decomposition. Therefore, besides providing experimental evidence for the effectiveness of Box-MPLP, we also seek to demonstrate the overall effectiveness of using a box-constrained objective for soft dual decomposition as an alternative to variable copying, regardless of what inference algorithm is used for

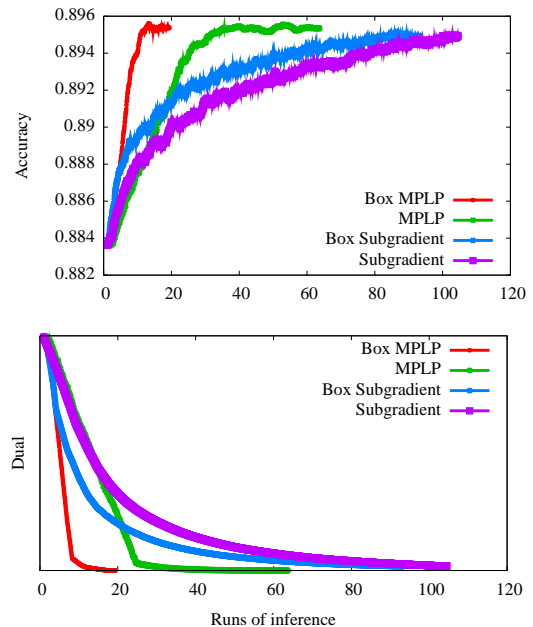
minimizing the box-constrained objective. Finally, note that all algorithms provide an $O(\frac{1}{\sqrt{t}})$ convergence rate, so they can only be compared empirically.

We mirror the experimental setup of Rush *et al.* (2012) for both tagging and parsing. To measure the speed of the algorithms, we record the total number of calls to inference in sentence-level problems, which we normalize by the number of sentences in the corpus to facilitate comparison across experiments. After the first pass, we only perform inference when relevant dual variables change.

Measuring inference calls rather than wall-clock time yields a more reliable experimental setting for the following two reasons: (1) it is independent of the implementation used, and (2) it allows us to be generous to the baseline algorithms we seek to outperform. First, we ignore the cost of running MAP inference in the tree-structured auxiliary problem in Figure 2. Second, we assign a pessimistic multiplier of two for all inference calls that require max-marginals. For NLP models with millions of features, this is an exaggeration because computing the model’s score vector w is typically the most costly step.

10.1 POS TAGGING

Figure 3: Accuracy (top) and dual objective (bottom) v.s. runs of sentence-level inference for WSJ-200 POS tagging.



Following Rush *et al.* (2012), we learn models on subsets of 50, 100, 200, and 500 sentences from the first chapter of the Penn Treebank and test on the Penn Treebank chapters test set (Marcus *et al.*, 1993). We use a bigram CRF tagger (Lafferty *et al.*, 2001). For all experiments, we report average sentence-level accuracy and the gains we obtain from corpus-wide inference in Appendix 6. Both are consistently comparable to Table 4 of Rush *et al.* (2012).

Table 1: Normalized number of inference runs for each algorithm to attain quantiles of the best dual solution in the WSJ-200 tagging experiment. If a quantile was not reached during 100 iterations, we show ‘na’.

Accuracy quantile	80%	85%	90%	95%
Subgradient	70	92	na	na
MPLP	22	23	25	30
Box-Subgradient	20	35	40	54
Box-MPLP	8	9	10	10
Dual Quantile	80%	85%	90%	95%
Subgradient	24	34	56	na
MPLP	21	22	23	35
Box-Subgradient	30	35	40	54
Box-MPLP	7	7	8	9

We present results from where we train on 200 sentences, but they are representative of the others, given in Appendix 6.1. Figure 3 shows the corpus-wide tagging accuracy and dual objective as a function of the sentence-level MAP calls. Recall that we double-count all calls to max-marginal routines. Table 1 shows how much inference is necessary to reach various percentile gains in accuracy and percentile reductions in the dual objective. Box-MPLP substantially outperforms both Box-Subgradient and MPLP, and the box-constrained versions of both algorithms outperform their variable-copying-based counterparts. Compared to the baseline subgradient algorithm used by Rush *et al.* (2012), we require 10x fewer MAP calls.

10.2 DEPENDENCY PARSING

Table 2: Iteration costs for the parsing experiments.

PTB to QTB				
Accuracy quantile	80%	85%	90%	95%
Subgradient	4.1	4.3	5.2	6.1
MPLP	4.3	4.3	4.3	‘na’
Box-Subgradient	2.1	2.1	2.4	2.8
Box-MPLP	2.6	2.8	3	‘na’
Dual quantile	80%	85%	90%	95%
Subgradient	3.0	3.2	3.4	3.9
MPLP	4.2	4.4	4.9	4.9
Box-Subgradient	1.6	1.7	1.8	2.0
Box-MPLP	2.5	2.5	2.5	2.6
QTB to PTB				
Dual quantile	80%	85%	90%	95 %
Subgradient	15	16	18	22
MPLP	14	15	16	17
Box-Subgradient	8.1	9.2	10	12
Box-MPLP	6.9	7.4	7.9	8.6

Our corpus-wide parsing experiments present a characteristically different regime for comparing the four algorithms because the graph of connections between the subproblems is much more sparse and the overall number of necessary iterations for the algorithms to converge is much lower.

Following Rush *et al.* (2012), each set of POS tags around a token defines a context, and identical contexts are encour-

aged to have parents with similar POS tags by introducing various consensus structures. We mirror their domain adaptation experiments, training on the Penn Treebank (PTB) and testing on the Question Treebank (QTB), and vice-versa (Judge *et al.*, 2006). We parse with a first-order projective arc-factored parser (McDonald *et al.*, 2005) using dynamic programming for inference, which has lower accuracy than the second-order projective parser used in Rush *et al.* (2012). Table 2 summarizes our results.

In the PTB-to-QTB experiment, the box-constrained algorithms uniformly outperform their counterparts based on variable copying. Unlike our POS experiments, however, Box-MPLP does not outperform Box-Subgradient. Since all the algorithms converge so quickly, the extra computation to obtain max-marginals is too costly (in the factor-2 scheme). Box-MPLP is still about 2x faster than Subgradient, which is what Rush *et al.* (2012) used, though. For the QTB-to-PTB experiment we were unable to reproduce accuracy increases as reported in Rush *et al.* (2012); none of the optimization algorithms managed to improve the accuracy for any setting of the penalties. This is probably due to our simpler parser. However, regarding dual optimization, each coordinate descent method outperforms its corresponding subgradient method, and the boxed algorithms outperform their variable-copying alternatives. Again, Box-MPLP was about 2x faster than Subgradient. See Appendix 6.2 for accuracy and dual figures.

11 CONCLUSION AND FUTURE WORK

Soft constraints can be easily modeled by imposing box constraints on an associated dual decomposition objective. This yields fast, simple-to-implement algorithms. Box-MPLP, a block coordinate descent algorithm, provides a competitive alternative to projected subgradient descent.

Future work will explore ways to adapt the alternative message passing algorithms discussed in Section 9 to handle box constraints and consider additional combinatorial factors besides soft constraints that can be ‘optimized out’ by imposing constraints in an associated dual problem.

12 ACKNOWLEDGMENT

This work was supported in part by the Center for Intelligent Information Retrieval, in part by DARPA under agreement number FA8750-13-2-0020 and in part by NSF grant #CNS-0958392. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

References

- Anzaroot, Sam, Passos, Alexandre, Belanger, David, & McCallum, Andrew. 2014. Learning Soft Linear Constraints with Application to Citation Field Extraction. *In: Proceedings of the 52th Annual Meeting of the Association for Computational Linguistics*.
- Chang, Ming-Wei, Ratnoff, Lev, & Roth, Dan. 2012. Structured learning with constrained conditional models. *Machine learning*, **88**(3), 399–431.
- Chieu, Hai Leong, & Teow, Loo-Nin. 2012. Combining local and non-local information with dual decomposition for named entity recognition from text. *Pages 231–238 of: 15th International Conference on Information Fusion*.
- Cortes, Corinna, & Vapnik, Vladimir. 1995. Support-vector networks. *Machine learning*, **20**(3), 273–297.
- Duchi, John, Tarlow, Daniel, Elidan, Gal, & Koller, Daphne. 2007. Using Combinatorial Optimization within Max-Product Belief Propagation. *Pages 369–376 of: Advances in Neural Information Processing Systems 19*.
- Finkel, Jenny Rose, Grenager, Trond, & Manning, Christopher. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. *Pages 363–370 of: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*.
- Globerson, Amir, & Jaakkola, Tommi. 2007. Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. *Advances in Neural Information Processing Systems*, **21**(1.6).
- Judge, John, Cahill, Aoife, & Van Genabith, Josef. 2006. Questionbank: Creating a corpus of parse-annotated questions. *Pages 497–504 of: Proceedings of the 21st International Conference on Computational Linguistics*.
- Komodakis, Nikos, Paragios, Nikos, & Tziritas, Georgios. 2007. MRF optimization via dual decomposition: Message-passing revisited. *Pages 1–8 of: IEEE 11th International Conference on Computer Vision*.
- Koo, Terry, Rush, Alexander M, Collins, Michael, Jaakkola, Tommi, & Sontag, David. 2010. Dual decomposition for parsing with non-projective head automata. *Pages 1288–1298 of: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*.
- Lafferty, John D, McCallum, Andrew, & Pereira, Fernando CN. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. *Pages 282–289 of: Proceedings of the Eighteenth International Conference on Machine Learning*.
- Luo, Zhi-Quan, & Tseng, Paul. 1992. On the convergence of the coordinate descent method for convex differentiable minimization. *Journal of Optimization Theory and Applications*, **72**(1), 7–35.
- Marcus, Mitchell P, Marcinkiewicz, Mary Ann, & Santorini, Beatrice. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational linguistics*, **19**(2), 313–330.
- Martins, Andre FT, Figueiredo, Mario AT, Aguiar, Pedro MQ, Smith, Noah A, & Xing, Eric P. 2012. Alternating directions dual decomposition. *arXiv preprint arXiv:1212.6550*.
- McDonald, Ryan, Crammer, Koby, & Pereira, Fernando. 2005. Online large-margin training of dependency parsers. *Pages 91–98 of: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*.
- Meshi, Ofer, Jaakkola, Tommi, & Globerson, Amir. 2012. Convergence Rate Analysis of MAP Coordinate Minimization Algorithms. *Pages 3023–3031 of: Advances in Neural Information Processing Systems 25*.
- Mezuman, Elad, Tarlow, Daniel, Globerson, Amir, & Weiss, Yair. 2013. Tighter Linear Program Relaxations for High Order Graphical Models. *In: Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence (UAI-13)*. Corvallis, Oregon: AUAI Press.
- Nesterov, Yurii. 2003. *Introductory lectures on convex optimization: A basic course*. Vol. 87. Springer.
- Ravikumar, Pradeep, Agarwal, Alekh, & Wainwright, Martin J. 2010. Message-passing for graph-structured linear programs: Proximal methods and rounding schemes. *The Journal of Machine Learning Research*, **11**, 1043–1080.
- Richtárik, Peter, & Takáč, Martin. 2012. Parallel coordinate descent methods for big data optimization. *arXiv preprint arXiv:1212.0873*.
- Rush, Alexander M., & Collins, Michael. 2012. A Tutorial on Dual Decomposition and Lagrangian Relaxation for Inference in Natural Language Processing. *J. Artif. Intell. Res. (JAIR)*, **45**, 305–362.
- Rush, Alexander M, Reichart, Roi, Collins, Michael, & Globerson, Amir. 2012. Improved parsing and pos tagging using inter-sentence consistency constraints. *Pages 1434–1444 of: Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Schwing, Alex, Hazan, Tamir, Pollefeys, Marc, & Urtasun, Raquel. 2012. Globally Convergent Dual MAP LP Relaxation Solvers using Fenchel-Young Margins. *Pages 2393–2401 of: Advances in Neural Information Processing Systems 25*.
- Smith, David A., & Eisner, Jason. 2008. Dependency Parsing by Belief Propagation. *Pages 145–156 of: Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Smith, Noah A. 2011. Linguistic structure prediction. *Synthesis Lectures on Human Language Technologies*, **4**(2), 1–274.
- Sontag, David, Globerson, Amir, & Jaakkola, Tommi. 2011. Introduction to Dual Decomposition for Inference. *In: Sra, Suvrit, Nowozin, Sebastian, & Wright, Stephen J. (eds), Optimization for Machine Learning*. MIT Press.
- Sutton, Charles, & McCallum, Andrew. 2006. *Introduction to statistical relational learning*. MIT Press. Chap. An introduction to conditional random fields for relational learning.
- Wainwright, Martin J, Jaakkola, Tommi S, & Willsky, Alan S. 2005. MAP estimation via agreement on trees: message-passing and linear programming. *Information Theory, IEEE Transactions on*, **51**, 3697–3717.
- Werner, Tomás. 2008. High-arity interactions, polyhedral relaxations, and cutting plane algorithm for soft constraint optimization (MAP-MRF). *In: CVPR 2008*.

Bayesian Interactive Decision Support for Multi-Attribute Problems with Even Swaps

Debarun Bhattacharjya and Jeffrey O. Kephart

Cognitive Computing Research, IBM T. J. Watson Research Center
1101 Kitchawan Rd, Rt. 134, Yorktown Heights, NY 10598, USA
E-mail: debarunb, kephart@us.ibm.com

Abstract

Even swaps is a method for solving deterministic multi-attribute decision problems where the decision maker iteratively simplifies the problem until the optimal alternative is revealed (Hammond et al. 1998, 1999). We present a new practical decision support system that takes a Bayesian approach to guiding the even swaps process, where the system makes queries based on its beliefs about the decision maker’s preferences and updates them as the interactive process unfolds. Through experiments, we show that it is possible to learn enough about the decision maker’s preferences to measurably reduce the cognitive burden, i.e. the number and complexity of queries posed by the system.

1 INTRODUCTION

In deterministic multi-attribute problems, the decision maker (or DM, for short) chooses among N alternatives, each of which has M attributes. An alternative \mathbf{x} is a vector of consequences for each attribute:

$$\mathbf{x} = \{x_i : i = 1, \dots, M\}, \quad (1)$$

where x_i is the consequence for attribute i . This is often represented as a *consequence table* such as the one illustrated in Fig 1(a), which displays alternatives and attributes for a hiring problem along its columns and rows respectively.

The DM’s preferences for the various attributes can be modeled using a *value function* $v(\mathbf{x})$. Additive value functions are a popular choice, mainly due to the ease with which they can be elicited:

$$v(\mathbf{x}) = \sum_{i=1}^M w_i v_i(x_i), \quad (2)$$

where attribute weights $\mathbf{w} = \{w_i : i = 1, \dots, M\}$ are non-negative and sum to 1 and the $v_i(x_i)$ represent one-dimensional marginal value functions. Note that we make a distinction between value and utility functions, following Keeney and Raiffa (1976), who reserve the term ‘utility function’ to characterize preferences under uncertainty.

There are several well-known approaches to eliciting additive value functions. The most popular ones tackle direct elicitation, where the DM reveals their trade-offs by answering questions pertaining to the weights and marginal value functions. von Winterfeldt and Edwards (1986) and Belton and Stewart (2002) review some well known weighting techniques. An alternate approach is that of *even swaps*, which is an indirect preference elicitation method that simultaneously solves a specific decision problem (Hammond et al. 1998, 1999). Here the DM answers a few simple and pointed queries to iteratively reduce the number of columns and rows in the consequence table until the optimal alternative is revealed. Mustajoki and Hämäläinen (2005, 2007) coined the term *smart swaps* to refer to guided even swaps, i.e. using a decision support system to provide process suggestions.

In this paper, we propose a Bayesian approach to guiding the even swaps process, whereby the system makes queries based on its beliefs about the DM’s preferences and updates them as the interactive process unfolds. The literature on Bayesian techniques for learning a DM’s preferences is vast and varied, spanning domains such as management science, artificial intelligence, expert systems and machine learning (e.g. Eliashberg and Hauser 1985, Jimison et al. 1992, Poh and Horvitz 1993, Chajewska et al. 2000, Anderson and Hobbs 2002, Boutilier 2002, Scott and Shachter 2005).

We review the even swaps method in Section 2. The subsequent three sections summarize our main contributions. In Section 3, we present some properties of even swaps, providing conditions under which they are feasible. In Section 4, we describe our swaps algorithm

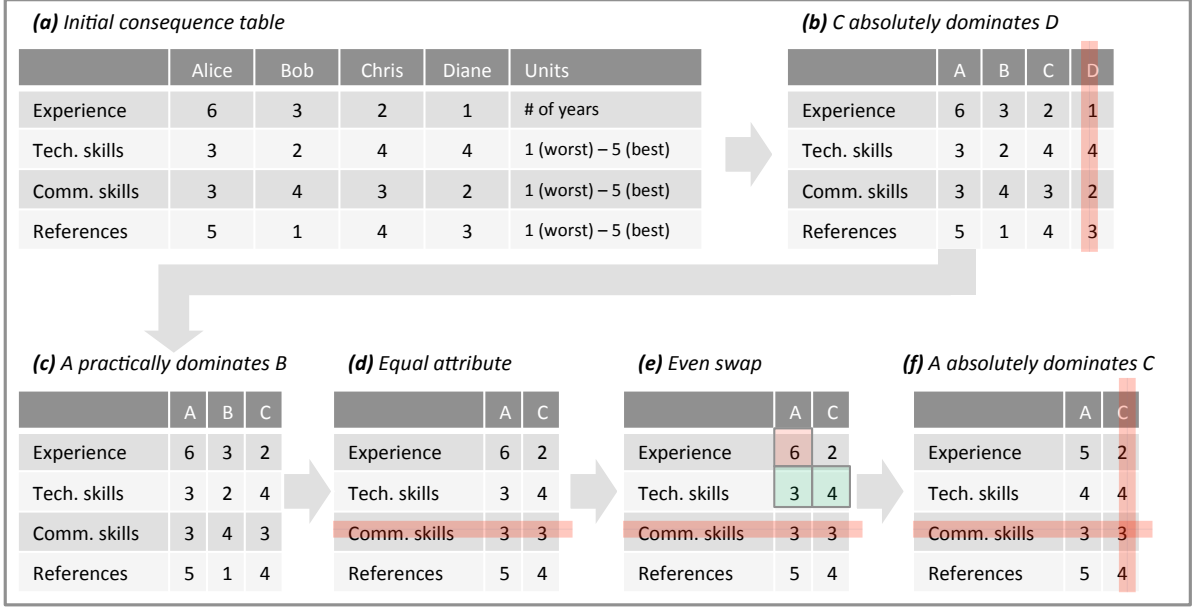


Figure 1: The even swaps method applied to a hiring problem.

in detail. In Section 5, we discuss the results of some experiments that study the effect of problem size and Bayesian learning on the number and type of queries made to the DM. We are not aware of any previous work with computer experiments that explores the effect of smart swaps on a set of consequence tables and DMs. Finally, we conclude in Section 6.

2 EVEN AND SMART SWAPS

We will explain the even swaps method with the help of the following illustrative example:

A Hiring Example. Figure 1(a) presents the consequence table for a manager Zoe who faces a hiring decision and must choose among four candidates — Alice, Bob, Chris and Diane — across four attributes: Experience (in # of years) and qualitative measures such as Technical Skills, Communication Skills and References, all scored on a scale of 1 (worst) to 5 (best).

Zoe chooses to pursue the even swaps method to determine the optimal hire. First, she recognizes that Chris scores at least as well as Diane on all attributes, and therefore removes Diane from consideration in 1(b). This is an example of *absolute dominance*. Next, she observes that Alice fares better on most attributes as compared to Bob, except for Technical Skills where Bob scores 1 point higher. Feeling that Alice compensates for this deficit along the other attributes, i.e. that Alice exhibits *practical dominance* over Bob, Zoe removes Bob from consideration in 1(c).¹

¹The original even swaps literature introduced practical

Zoe then notices that the remaining candidates, Alice and Chris, have the same score (3) on Communication Skills. Reasoning that she need not be concerned with this attribute in subsequent iterations, as she can make subsequent value judgments conditional on this common score, she greys this attribute out in 1(d). In the even swaps literature, this task is referred to as identifying an irrelevant attribute; for reasons explained in the next section, we prefer the term *equal attribute*, and say that this attribute has become inactive.

Now Zoe makes the move that gives the *even swap* method its name. She observes that Alice fares worse on Technical Skills, but better on the remaining active attributes. She answers the following question, indicated by the three boxes in 1(e): how many years of Experience would she be willing to give up for Alice to improve her Technical Skills score from 3 to 4? An even swap produces a hypothetical equivalent alternative in which a change in the consequences of one attribute balances the change in the consequences of another, and is a specific kind of matching query (Delqu   1993). The DM’s response is determined by her value judgments; in this case, she determines that Alice’s Experience should change from 6 to 5 years. She then replaces Alice with her hypothetical clone in the consequence table in 1(f). In the final step, she recognizes that Alice absolutely dominates Chris, thereby revealing Alice to be the optimal candidate. □

dominance as an intuitive but vague notion. Subsequent work on smart swaps proposed a definition with some practical drawbacks. A major contribution of our work is a precise definition and demonstration of its practicality.

The key idea that differentiates indirect methods like even swaps from direct elicitation techniques is that the analyst/system need not have a complete picture of the DM’s preferences to find the optimal alternative for a particular decision. It is therefore often beneficial to use such techniques for reducing elicitation burden and potential inaccuracies, as people are highly susceptible to cognitive biases (Lichtenstein and Slovic 2006).

The even swaps method appears to be suitable for small problems where the interactive nature of the method, the access to the alternatives and the (almost) instant gratification from solving the problem appeal to the DM. It is particularly useful for DMs who either find it difficult to answer questions about their trade-offs in terms of weight ratios, or who need to view/consider the alternatives to construct their preferences. Kajanus et al. (2001) provide an application to strategy selection in rural enterprises.

Even swaps was originally intended to be self-guided; Mustajoki and Hämäläinen (2005, 2007) propose a decision support system for smart swaps using preference programming, i.e. by recognizing the feasible region of weights for fixed bounds on marginal value functions. Their model makes the practical dominance notion precise by recommending it through pairwise dominance, which occurs when there is no way an alternative can be most preferred, based on the feasible weight region and bounds. Their method however has several limitations. For instance, there is little the system can do if it proposes a practical dominance query and the DM rejects it, aside from changing bounds midway through the process. Crucially, they are unable to recognize swaps that are not feasible.

Here we propose a Bayesian approach that exploits prior information about the feasible weight region as represented by a prior probability distribution. We introduce the notion of probable dominance as well as a heuristic that recommends even swaps through probabilistic computations. The system easily handles rejection of practical dominance queries. We also present new results about feasibility conditions for even swaps, using them to recognize and adapt to declarations of infeasible swaps. Our approach is particularly adept at providing inexperienced users with specific recommendations. However, as discussed in the conclusions, our method possesses its own set of limitations.

3 PROPERTIES OF EVEN SWAPS

The overarching even swaps method gets its name from the even swap query, which is crucial towards reducing the size (and therefore complexity) of the consequence table. In this section, we specify our assumptions for the class of multi-attribute problems under considera-

tion, and then present some results pertaining to the properties of even swaps.

3.1 ASSUMPTIONS

We address multi-attribute problems where the DM has an additive value function, i.e. of the form in equation (2). This is applicable only when attributes are mutually preferentially independent. As noted by previous authors, this is a common assumption and is widely applied in practice (Keeney and Raiffa 1976, Stewart 1996, Belton and Stewart 2002).

In theory, the even swaps method is applicable for all value functions and is not restricted to the additive form. However, the method can be challenging to apply when there is value dependence, in which case the DM would have to consider consequence levels of all attributes while making a judgment about an even swap. In that sense, no attribute would be ‘irrelevant’ when the DM makes the even swap based on their trade-offs. It is difficult to imagine the method being implemented successfully in such a situation without an analyst in the room to guide the DM. The additive assumption therefore makes an automated decision support system more likely to be used (and perhaps misused).

We also assume that the one-dimensional marginal value functions are continuous, bounded and monotonic. Since they are bounded, these functions can be normalized such that $0 \leq v_i(x_i) \leq 1$, $v_i(x_i^0) = 0$ and $v_i(x_i^*) = 1$ for all attributes, where x_i^0 and x_i^* represent the least and most preferred consequences for attribute i . The domain of an attribute is denoted D_i , therefore for an attribute where more is preferred to less, $D_i = [x_i^0, x_i^*]$. Monotonic attributes are common in practice; non-monotonic attributes can sometimes be redefined so as to render them monotonic. Furthermore, a discrete attribute can often be approximated as continuous. For instance, in the hiring problem in Figure 1, three of the four attributes are measured as integers on a scale of 1 to 5, but they could easily be approximated as continuous attributes. These assumptions are therefore not too restrictive.

3.2 NOTATION AND PROPERTIES

The even swaps method attempts to guide the DM by simplifying the consequence table. During this interactive process, the DM must carefully consider pairs of alternatives and their consequences along specific attributes. A consequence x_i is deemed to be preferred over y_i if it has higher marginal value:

$$x_i \succ y_i \Leftrightarrow v_i(x_i) > v_i(y_i). \quad (3)$$

Any pair of alternatives \mathbf{x} and \mathbf{y} can therefore be associated with the following three sets of attributes:

dominating set $D(\mathbf{x}, \mathbf{y}) = \{i : x_i \succ y_i\}$, **non-dominating set** $N(\mathbf{x}, \mathbf{y}) = \{i : x_i \prec y_i\}$, and **equal set** $E(\mathbf{x}, \mathbf{y}) = \{i : x_i = y_i\}$. Note that $N(\mathbf{x}, \mathbf{y}) = D(\mathbf{y}, \mathbf{x})$.

The task with perhaps the lowest cognitive load for the DM and the lowest computational load for a system is identifying equal attributes. While somewhat more complex for a DM, it is also trivial for a system to discover absolute dominance, denoted $\mathbf{x} \succeq^A \mathbf{y}$, using non-dominating attribute sets:

$$\mathbf{x} \succeq^A \mathbf{y} \Leftrightarrow N(\mathbf{x}, \mathbf{y}) = \emptyset. \quad (4)$$

For a replicate pair of solutions, i.e. where both $N(\mathbf{x}, \mathbf{y}) = \emptyset$ and $D(\mathbf{x}, \mathbf{y}) = \emptyset$, either \mathbf{x} or \mathbf{y} can be removed from the table at random.

Practical dominance comes under consideration when one of the sets $N(\mathbf{x}, \mathbf{y})$ and $D(\mathbf{x}, \mathbf{y})$ has many more elements than the other. While practical dominance claims help remove some solutions, the DM may eventually have to perform an even swap to manipulate the consequence table and make further progress. We denote an even swap as $s(x_i \rightarrow x'_i, x_j \rightarrow x'_j)$, where the alternative \mathbf{x} is modified by the DM, such that the change from x_i to x'_i along attribute i is compensated by the change from x_j to x'_j along attribute j .

Consider the even swap in the hiring example, where the DM provided a response to a change from the score 3 to 4 on Technical Skills, along Experience. Note that the swap performed was specifically designed to make consequences identical for Technical Skills. This type of swap is relatively cognitively comfortable for the DM, since they are able to observe the numbers along a specific row. Moreover, ensuring equal consequences simplifies the table and allows for potential ease of elicitation in future tasks. We refer to such a swap as an **equalizing even swap**, defined as an even swap that makes the consequences of two alternatives equal along an attribute. For any two alternatives \mathbf{x} and \mathbf{y} , $s(x_i \rightarrow y_i, x_j \rightarrow x'_j)$ is an equalizing even swap because it makes attribute i 's consequences for both alternatives equal, thereby increasing the set $E(\mathbf{x}, \mathbf{y})$.

Is an even swap always possible? No. The following proposition provides the conditions under which an even swap is feasible, assuming that the DM's response is consistent with their value function.

Proposition 1 (Even Swap Feasibility). *The even swap $s(x_i \rightarrow x'_i, x_j \rightarrow x'_j)$, $i \neq j$, $x_i, x'_i \in D_i$ is feasible only if:*

- (i) When $x'_i \succ x_i$: $v_j(x_j) \geq \frac{w_i}{w_j} [v_i(x'_i) - v_i(x_i)]$
- (ii) When $x'_i \prec x_i$: $1 - v_j(x_j) \geq \frac{w_i}{w_j} [v_i(x_i) - v_i(x'_i)]$

Proof. If $x'_i \succ x_i$, the swap is not feasible when even a response of $x'_j = x_j^0$ cannot compensate for the change, which occurs when $w_j [v_j(x_j) - v_j(x_j^0)] < w_i [v_i(x'_i) - v_i(x_i)]$. The result follows after recognizing $v_j(x_j^0) = 0$. The other case is similar. \square

The fact that not all swaps are feasible is potentially problematic for a system attempting to guide the process by recommending equalizing even swaps. Since the system is not exactly aware of the DM's preferences during the process, it is possible for the system to propose a swap that is infeasible for the DM. Fortunately, as determined in the following proposition, if the swap $s(x_i \rightarrow y_i, x_j \rightarrow x'_j)$ is not feasible, its **conjugate swap** $s(x_j \rightarrow y_j, x_i \rightarrow x'_i)$ must be feasible.

Proposition 2 (Equalizing Even Swap Feasibility).

For any two alternatives \mathbf{x} and \mathbf{y} that do not dominate each other over attributes i and j , at least one of the equalizing even swaps $s(x_i \rightarrow y_i, x_j \rightarrow x'_j)$ or $s(x_j \rightarrow y_j, x_i \rightarrow x'_i)$ is feasible.

Proof. Suppose that $y_i \succ x_i$. If the swap $s(x_i \rightarrow y_i, x_j \rightarrow x'_j)$ is not feasible, then from Proposition 1(i), $v_j(x_j) < \frac{w_i}{w_j} [v_i(y_i) - v_i(x_i)]$. Rearranging, $w_i v_i(x_i) + w_j v_j(x_j) < w_i v_i(y_i)$. For the conjugate swap, by definition, $w_i v_i(x_i) + w_j v_j(x_j) = w_i v_i(x'_i) + w_j v_j(y_j)$. Using the condition from the infeasibility of the original swap, $w_i v_i(x'_i) + w_j v_j(y_j) < w_i v_i(y_i) \implies w_i v_i(x'_i) < w_i v_i(y_i) \implies x'_i \prec y_i$. The conjugate swap is therefore indeed feasible. The other case is similar. \square

The implication of these results is that a feasible swap can always be found: if the DM declares that a given even swap is infeasible, then the conjugate swap will be feasible, and the system can recommend it.

Note that both propositions assume that a DM's response is consistent with their value function. However, behavioral research on bi-matching suggests people may provide inconsistent responses between queries pertaining to a swap vs. its conjugate (Delqu   1997, Willemsen and Keren 2003). In the algorithm described in the next section, we assume the DM is willing to make either a swap or its conjugate, but we allow for noise in the response to an even swap query.

4 BAYESIAN SMART SWAPS

In our formulation of guiding an interactive even swap, we assume the system has prior beliefs $p(\mathbf{w})$ about the DM's weights in their additive value function. If there is no a priori information available, the system may choose a uniform prior over the weight simplex: $p(\mathbf{w}) \sim \text{Dirichlet}(\boldsymbol{\alpha})$ where $\boldsymbol{\alpha}$ is a vector of 1s.

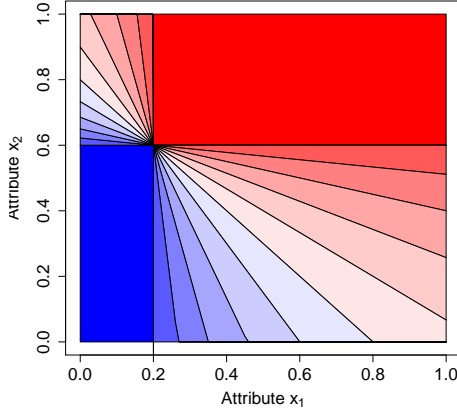


Figure 2: Regions of absolute and practical dominance for the two-attribute example when $w_1 \sim \text{Uniform}(0, 1)$.

We also assume for now that the system knows the DM’s marginal value functions, perhaps through prior assessments. Since these are one-dimensional functions, they are usually easier to elicit than weights that reflect trade-offs. In sub-section 4.4 we briefly outline how our algorithm may be extended to the case of unknown marginal functions.

We explore how a system can cope with uncertainty about the DM’s weights, incorporating responses to recommended practical dominance and even swap queries from the DM. In our algorithm, the system gradually learns the user’s preferences and exploits it for the sole purpose of reaching the optimal alternative as soon as possible. In the following sub-sections, we describe various aspects of our overall approach.

4.1 ABSOLUTE VS. PROBABLE DOMINANCE

One of the central notions of the original even swaps method is that of practical dominance, according to which an alternative can be discarded if it appears to be *nearly* absolutely dominated by another. In this section, we view practical dominance through a Bayesian lens, with the intent of reducing the cognitive burden of DMs. To motivate our approach, let us first study absolute dominance.

Consider an alternative \mathbf{x} whose consequences have been normalized; therefore it lies somewhere in the unit cube. \mathbf{x} dominates a proportion of other alternatives given by the volume $\prod_{i=1}^M x_i$, and is dominated by a proportion $\prod_{i=1}^M (1 - x_i)$. Note that if a family of problems is built by generating alternatives uniformly over the consequence domains, then the probability that any particular alternative dominates another decreases exponentially with the number of attributes M . Therefore absolute dominance does not occur with sufficient frequency to be a basis for a practical decision support algorithm. Moreover, in real-world set-

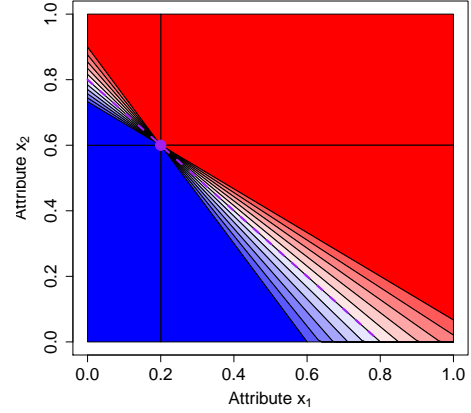


Figure 3: Regions of absolute and practical dominance for the two-attribute example when $w_1 \sim \text{Uniform}(0.4, 0.6)$.

tings, absolutely dominated alternatives would likely be shelved before reaching the conference room.

A relationship that might be more useful is that of **probable dominance**, which measures the system’s beliefs about whether the DM prefers an alternative to another. The probability that alternative \mathbf{x} dominates \mathbf{y} is denoted $p_{\mathbf{x}\mathbf{y}}$, where:

$$p_{\mathbf{x}\mathbf{y}} = \int_{\mathbf{w}} \left(\sum_{i=1}^M w_i [v_i(x_i) - v_i(y_i)] \geq 0 \right) p(\mathbf{w}) d\mathbf{w}. \quad (5)$$

If the system believes that the DM is likely to prefer an alternative over another, perhaps it can recommend them as a candidate pair for practical dominance. Although the DM makes the eventual judgment, the system recommends the pair in the hope that it will simplify the problem. We therefore propose probable dominance above a certain threshold p_T to recognize potential practical dominance, $p_{\mathbf{x}\mathbf{y}} \geq p_T$.

Let us study the following simple example to compare the occurrence of absolute and probable dominance:

A Two-attribute Example. Suppose $M = 2$ and that the DM’s marginal value functions are linear and normalized to between 0 and 1. As a reference, suppose that the DM’s trade-offs are accurately captured by weights $w_1 = 0.5$ and $w_2 = 0.5$.

Figure 2 illustrates the regions of absolute and practical dominance with respect to a chosen alternative $\mathbf{x} = (0.2, 0.6)$ (represented as a purple dot) when the system believes that $w_1 \sim \text{Uniform}(0, 1)$. Alternatives that absolutely dominate \mathbf{x} are shown in dark red, while those that are absolutely dominated by \mathbf{x} are shown in dark blue. The regions of potential practical dominance (as determined by probable dominance) for various values of the probability p appear as bands of lighter red and blue, in increments of 0.1, ranging from $p = 0.9$ to 1.0 (almost deep red) down to $p = 0$ to 0.1 (almost deep blue).

Figure 3 illustrates almost the same situation except now the system believes that $w_1 \sim \text{Uniform}(0.4, 0.6)$, possibly by learning from responses to previous queries. It is immediately apparent that the reduced uncertainty yields enlarged regions in which there is a high certainty that \mathbf{x} dominates or is dominated. These regions now appear as large triangles flanking the rectangular regions of absolute dominance. From a practical perspective, such regions are effectively equivalent to those of absolute dominance.

Relative to Fig. 2, the region of uncertainty is much more tightly clustered around the diagonal line $x_1 + x_2 = 0.8$ that represents the boundary between $x \succ y$ and $y \succ x$. This is due to the reduced uncertainty about the true value of \mathbf{w} , and illustrates the benefits of reducing the uncertainty: it allows the system to be more confident in suggesting potential practical dominance to the DM. \square

The simple example illustrates that a pair of alternatives chosen at random is more likely to exhibit probable rather than absolute dominance, making it more useful in practice. Further numerical simulations were performed, demonstrating that the probability for a given vector \mathbf{x} to practically dominate a given vector \mathbf{y} above a given threshold p_T is insensitive to the number of attributes M . This suggests that probable dominance may be a useful concept in practice.

Algorithm 1 summarizes the system’s approach to recommending practical dominance and updating beliefs. We assume that the DM responds accurately to this query based on their preferences, since this is a comparison question that is typically associated with low cognitive load. This implies that the polytope of the weight region can be updated to incorporate the following condition:

$$\sum_{i=1}^M w_i [v_i(x_i) - v_i(y_i)] \geq (\leq) 0, \quad (6)$$

depending on whether the user responds ‘yes’ or ‘no’ to the question: do you prefer \mathbf{x} over \mathbf{y} ? The implications and potential limitations of assuming an accurate response to this query are discussed in the conclusions.

4.2 EVEN SWAPS

Recommending an effective even swap is more challenging than computing practical dominance. In Section 3, we explored some properties, discovering that not all swaps are feasible. The notion of an equalizing even swap was introduced as a practical means of forming a simpler consequence table. To make an equalizing even swap $s(x_i \rightarrow y_i, x_j \rightarrow x'_j)$, the system needs an alternative pair \mathbf{x}, \mathbf{y} and an attribute pair i ,

Algorithm 1 Practical Dominance Query

Input: N alternatives, threshold p_T , prior $p(\mathbf{w})$
Initialize $p_D^{max} = 0$
for each pair of vectors \mathbf{x} and \mathbf{y} **do**
 Compute p_{xy} from equation (5)
 if $p_{xy} \geq \max(p_T, p_D^{max})$ **then**
 Store pair \mathbf{x}, \mathbf{y} ; $p_D^{max} = p_{xy}$
 end if
end for
if $p_D^{max} \neq 0$ **then**
 Recommend potential practical dominance for \mathbf{x}, \mathbf{y} , inquiring whether $\mathbf{x} \succeq \mathbf{y}$
 Update $p(\mathbf{w})$ in accordance with DM’s response, using equation (6)
else
 There is no candidate pair
end if

j . Moreover, the system should be able to handle an infeasible swap.

We present a heuristic for recommending an even swap that identifies the most suitable alternative and attribute pairs. There are two main steps involved. In the first step, the system identifies alternatives \mathbf{x}, \mathbf{y} where it believes \mathbf{x} might be preferred over \mathbf{y} . It is natural to use probable dominance to quantify this belief. In the second step, the system identifies attributes $i \in N(\mathbf{x}, \mathbf{y})$ and $j \in D(\mathbf{x}, \mathbf{y})$ such that swap $s(x_i \rightarrow y_i, x_j \rightarrow x'_j)$ is likely to decrease $|N(\mathbf{x}, \mathbf{y})|$.

The intuition behind the heuristic is that an even swap query potentially pushes a pair of alternatives towards dominance of some sort, making it eventually evident to both the system and the DM. Focusing on a pair where one is likely to dominate the other and reducing the non-dominated attribute set ensures that this occurs. As shown in Section 3, an infeasible swap always possesses a feasible conjugate swap, so the heuristic is guaranteed to make progress (from a normative perspective). Note that the cognitive effort expended by the user in trying to respond to the original (failed) swap will be helpful in responding to its conjugate. Note also that the heuristic is myopic in that it attempts to find the ‘best’ swap at the current moment, without regard to long-term savings. It can be viewed as a dominance-focused heuristic, as it tries to drive alternative pairs towards dominance.

Suppose that the system is considering the equalizing even swap $s(x_i \rightarrow y_i, x_j \rightarrow x'_j)$ based on the aforementioned heuristic. By definition:

$$v_j(x'_j) = \frac{w_i (v_i(x_i) - v_i(y_i)) + w_j v_j(x_j)}{w_j}, \quad (7)$$

if it is feasible, i.e. satisfies Proposition 1.

Algorithm 2 Even Swap Query

Input: N alternatives, swap response noise δ , prior $p(\mathbf{w})$
Set threshold $p_T = 0$ and find alternative pair \mathbf{x}, \mathbf{y} from Algorithm 1
Initialize $p_S^{max} = 0$
for each pair of attributes i in $N(\mathbf{x}, \mathbf{y})$ and j in $D(\mathbf{x}, \mathbf{y})$ **do**
 Compute p_S from equation (8)
 if $p_S \geq p_S^{max}$ **then**
 Store pair i, j ; $p_S^{max} = p_S$
 end if
end for
Recommend the swap $s(x_i \rightarrow y_i, x_j \rightarrow x'_j)$
if Response is x'_j **then**
 Update $p(\mathbf{w})$ with conditions from equation (9)
else if DM declares swap is infeasible **then**
 Recommend conjugate swap $s(x_j \rightarrow y_j, x_i \rightarrow x'_i)$
 Update $p(\mathbf{w})$ using equation (9), after swapping i and j
end if

Suppose that i and j are both attributes where more is preferred to less. Then x_i is increased to y_i for the swap (because $i \in N(\mathbf{x}, \mathbf{y})$), therefore x_j is decreased to x'_j if the swap is feasible. The probability that this swap will decrease the non-dominated set p_S is:

$$\begin{aligned}
p_S &= P(x'_j \geq y_j) = P(v_j(x'_j) \geq v_j(y_j)) \\
&= \int_{\mathbf{w}} \left(\sum_{k=i,j} w_k [v_k(x_k) - v_k(y_k)] \geq 0 \right) p(\mathbf{w}) d\mathbf{w},
\end{aligned} \tag{8}$$

where the final step is a result of integration after rearranging (7). For the sake of brevity, we have notationally omitted specifying that p_S is a function of the swap; it should be inferred that it is associated with swap $s(x_i \rightarrow y_i, x_j \rightarrow x'_j)$. Also, note that although equation (8) applies only when i and j have monotonically increasing marginal value functions, it is easy to generalize it to include all other cases.

Algorithm 2 summarizes the system's approach to recommending even swaps and updating beliefs. Since an even swap is associated with a significant cognitive load, the system treats the response to lie within a noise band measured using the *swap response noise* δ . For instance, if a user responds to an even swap query with normalized consequence 0.6 and $\delta = 0.2$, then the system forms a lower bound $L_\delta = 0.5$ and upper bound $U_\delta = 0.7$. This noise band is subject to the other constraints posed on a response, i.e. it must lie within the domain. Therefore a response of 0.05 with $\delta = 0.2$ results in $L_\delta = 0$ and $U_\delta = 0.15$. If more of attribute j is preferred to less, the polytope

Algorithm 3 Bayesian Smart Swaps

Input: N alternatives, threshold p_T , swap response noise δ , prior $p(\mathbf{w})$
while more than 1 solution and 1 active attribute remain in table **do**
 Remove absolutely dominated solutions, if any
 Mark any attributes with equal consequences across alternatives as inactive, if any
 Identify potential practical dominance using Algorithm 1
 if practical dominance detected **then**
 Recommend it and update $p(\mathbf{w})$ from response
 else
 Recommend an even swap using Algorithm 2 and update $p(\mathbf{w})$ from response
 end if
end while
if single attribute remains **then**
 Find the optimal alternative \mathbf{x}
end if
Return \mathbf{x}

of the weight region can be updated with conditions from two inequalities involving w_i and w_j :

$$L_\delta \leq \frac{x'_j - x_j^0}{x_j^* - x_j^0} \leq U_\delta, \tag{9}$$

where L_δ and U_δ are bounds on normalized consequences ($L_\delta, U_\delta \in [0, 1]$) that depend on the DM's response and δ as described above, and x'_j is a function of the weights and marginal values as in equation (7). This equation could be easily modified for the case where less of attribute j is preferred.

4.3 HIGH-LEVEL ALGORITHM

Now that we have outlined the two main sub-routines – practical dominance and even swaps – Algorithm 3 provides the high-level routine for our proposed interactive even swaps method. The algorithm identifies absolute dominance and equal attributes, recommends practical dominance when it is confident enough, and recommends an equalizing even swap based on a dominance focused heuristic. The algorithm terminates when the optimal alternative is revealed.

4.4 EXTENSION: UNKNOWN MARGINAL VALUE FUNCTIONS

In the algorithm described here, we assumed that the system already knew the marginal value functions, perhaps through initial assessments. There is a natural extension to the case of unknown marginal value functions along the lines of Mustajoki and Hämäläinen

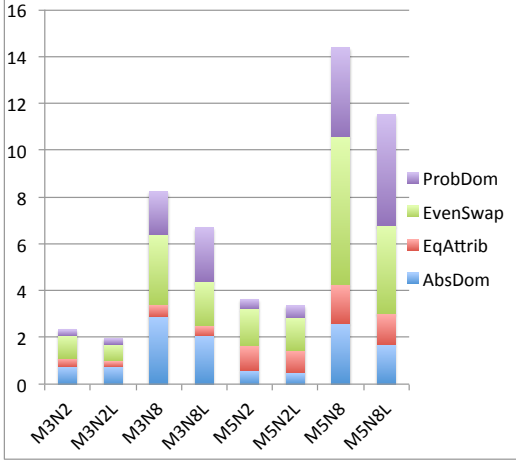


Figure 4: The effect of learning upon the number and type of queries and events. Average number of absolute dominance and equal attribute events, as well as probable dominance and even swap queries per scenario, for $M = \{3, 5\} \times N = \{2, 8\}$, with learning turned on (L) and off.

(2005, 2007), using previously determined lower and upper bounds on the marginal value functions. Subsequently, for all probabilistic computations — in this case those pertaining to computing probable dominance and the probability that the swap will decrease the non-dominated set — the system could use probability bounds for making recommendations and update its beliefs based on inequalities from these bounds. The current algorithm could be updated to incorporate these changes.

5 EXPERIMENTAL RESULTS

A first set of experiments were conducted to assess the degree to which learning reduces the number and complexity of queries directed to the DM. The high-level algorithm described in Section 4 was applied to a set of 100 randomly generated scenarios, each involving a randomly generated set of N alternatives with M attributes. Each of the NM values in the consequence table was generated from a Uniform distribution (0, 1). The user’s true weights were drawn uniformly from the $(M - 1)$ -dimensional unit simplex, and the prior was the same uniform distribution over the simplex. For simplicity, the marginal value functions were assumed to be linear and ranging from 0 to 1.

For each scenario, the probability threshold for a probable dominance query was set relatively high (0.9) to ensure that the queries might not be too onerous for real humans to answer. The probability that $\mathbf{x} \succ \mathbf{y}$ for the DM was computed by randomly generating at least 10000 weight vectors uniformly in the $(M - 1)$ -dimensional unit simplex. First, rejection sampling

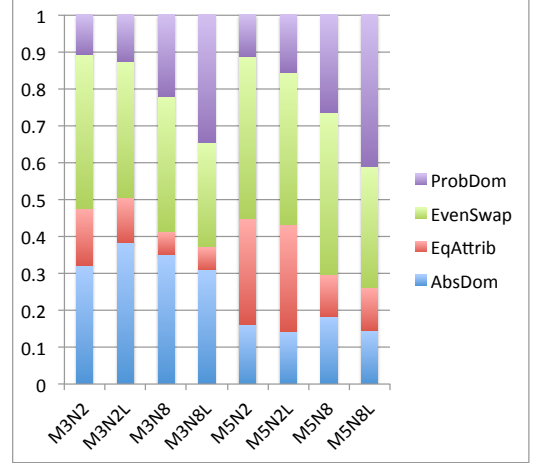


Figure 5: Same data as in Figure 4, except that the scales are normalized to 1 to illustrate the relative contributions of the different types queries and events.

was employed, i.e. the randomly generated weight vectors were reduced to a set that satisfied any constraints introduced during the interactive process. Then the probable dominance probability was computed as the fraction of weight vectors for which $\mathbf{x} \succ \mathbf{y}$. Particularly in cases where several even swaps had been applied, the weights were pinned down so precisely that the number of samples satisfying the constraints dropped below 100, in which case more points were generated to ensure that the probable dominance probability was computed from reasonable statistics. To simulate the DM answering a probable dominance or even swaps query, the true weight vector was used to generate the response that the DM would have generated. The DM’s noise about the swap value was modeled using a modest swap response noise of $\delta = 0.2$.

For each scenario, the number of absolute dominance and equal-attribute events (accomplished purely through system computations) were recorded. The number of probable dominance and even swap queries (including both regular and conjugate swaps) were recorded as well; these are queries that must be answered by the DM and therefore entail some cognitive burden. Eight sets of 100 scenarios were run, with the number of attributes set to $M = \{3, 5\}$, the number of solutions set to $N = \{2, 8\}$, and the method’s learning element both turned on and turned off. The results are summarized in Figure 4.

For the smallest scenarios ($(M, N) = (3, 2)$), an average of just two queries and/or events is required, and typically there is one absolute dominance event and one even swap, with probable dominance and elimination by virtue of equal attributes playing a relatively minor role. Due to the small number of queries and/or

events, learning has little impact. The average number of queries and/or events decreases from 2.33 ± 0.11 to 1.96 ± 0.11 — a drop that is of marginal statistical significance. On the other hand, when the number of solutions is increased from 2 to 8, the average number of queries and/or events rises to 8.22 ± 0.34 without learning and 6.7 ± 0.23 with learning — a statistically significant decrease of 18%. When the number of attributes is increased from 3 to 5, a similar trend is observed. For $N = 2$ solutions, the number of queries and/or events is 3.63 ± 0.27 without learning and 3.35 ± 0.24 with learning — an insignificant difference — whereas for $N = 8$ solutions the number of queries and/or events is 14.37 ± 0.57 and 11.51 ± 0.41 — a statistically significant drop of 20%.

Figure 5 provides another view of the same data, in which the relative contribution of the various types of queries and events is obtained by normalization. As anticipated, the impact of absolute dominance decreases as the number of solutions N increases from 2 to 8. This is a consequence of the exponential decrease in the probability for any given vector to absolutely dominate another with the number of attributes. Another trend evident here is that as N increases, the relative impact of probable dominance queries grows stronger. Moreover, for larger problems, the effect of learning is to further increase the relative importance of probable dominance over even swap queries.

Having established that learning can substantially reduce the number of queries and/or events required to identify the optimal alternative, and moreover that it shifts the balance more from even swaps to probable dominance queries as the problem size grows, a second series of experiments were conducted with learning turned on. The objective of these experiments was to chart in greater detail how the number and type of queries and/or events change as the number of attributes and alternatives are varied. The results depicted in Figure 6 demonstrate the same basic trends, including the waning importance of absolute dominance as the number of attributes M grows and the ascendancy of probable dominance as N grows.

6 CONCLUSIONS

In this paper, we have presented a method for guiding the DM through the even swaps process using an overall Bayesian approach with a dominance focused heuristic. We have demonstrated through experiments that one can effectively learn about the DM’s preferences in the course of a single session to guide them quickly to a final choice. A potential next step is to implement a tool and test its efficacy through experiments with real human subjects. Belton et al. (2005)

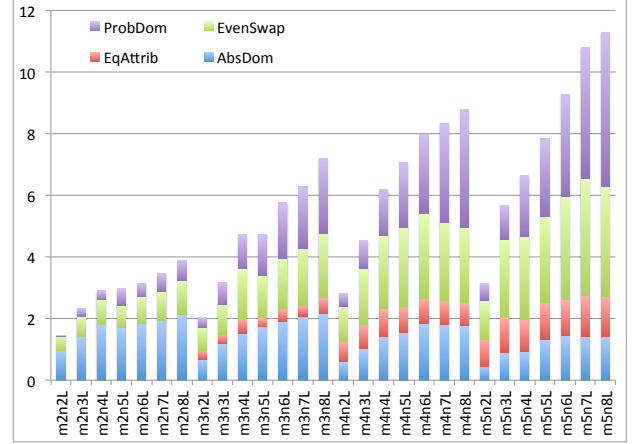


Figure 6: The effect of M and N on the number and type of queries and events. Average number of queries/events of each type, from left to right, for $M = \{2, 3, 4, 5\}$ and $N = \{2, 3, 4, 5, 6, 7, 8\}$.

conduct some user experiments involving even swap queries but such studies remain few and far between.

Our approach appears to be practical for modest-sized decision problems ($N < 10$). One could argue that direct elicitation techniques might be appropriate for large N (~ 100); however, if the DM prefers to use even swaps (for reasons highlighted in Section 2), it may be prudent to focus on learning the DM’s preferences rather than myopically trying to find the most likely pair of alternatives such that one might dominate the other.

Here we used a simple model for incorporating potential noise in a DM’s response to an even swap query; it was chosen to enable conditions represented as inequalities. In future research, we envision more nuanced noise models accounting for cognitive effects such as attribute conflict (e.g. Fischer et al. 2000, Delquí 2003). Also, although we have used probable dominance to measure practical dominance, it remains unclear how the metric would work for large problems because it may be difficult for a DM to compare any two arbitrary alternatives; note that comparison queries are also subject to various cognitive biases in general (e.g. Tversky et al. 1988, Tversky and Kahneman 1991). Finally, regarding the computations for simulation, rejection sampling is sufficient when only a few queries are asked in a single setting. If several queries are asked back-to-back, efficient methods such as hit and run sampling may be more effective.

Acknowledgments

We thank three anonymous reviewers for their valuable feedback.

References

- R. M. Anderson and B. F. Hobbs (2002) Using a Bayesian approach to quantify scale compatibility bias. *Management Science*, **48**(12):1555–1568.
- V. Belton and T. Stewart (2002) *Multiple Criteria Decision Analysis: An Integrated Approach*. Norwell, MA: Kluwer Academic Publishers.
- V. Belton, G. Wright and G. Montibeller (2005) When is swapping better than weighting? An evaluation of the even swaps method in comparison with multi attribute value analysis. Working Paper, 2005/19, Department of Management Science, University of Strathclyde, UK.
- C. Boutilier (2002) A POMDP formulation of preference elicitation problems. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI)*, 239–246, AAAI Press.
- U. Chajewska, D. Koller and R. Parr (2000) Making rational decisions using adaptive utility elicitation. In *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI)*, 363–369, AAAI Press.
- P. Delquié (1993) Inconsistent trade-offs between attributes: New evidence in preference assessment biases. *Management Science*, **39**(11):1382–1395.
- P. Delquié (1997) “Bi-matching”: A new preference assessment method to reduce compatibility effects. *Management Science*, **43**(5):640–658.
- P. Delquié (2003) Optimal conflict in preference assessment. *Management Science*, **49**(1):102–115.
- J. Eliashberg and J. Hauser (1985) A measurement error approach for modeling consumer risk preference. *Management Science*, **15**(1):1–25.
- G. W. Fischer, J. Jia and M. F. Luce (2000) Attribute conflict and preference uncertainty: The RandMAU model. *Management Science*, **46**(5):669–684.
- J. S. Hammond, R. L. Keeney and H. Raiffa (1998) Even swaps: A rational method for making trade-offs. *Harvard Business Review*, **76**(2):137–149.
- J. S. Hammond, R. L. Keeney and H. Raiffa (1999) *Smart Choices: A Practical Guide to Making Better Decisions*. Harvard Business School Press.
- H. B. Jimison, L. M. Fagan, R. D. Shachter and E. H. Shortliffe (1992) Patient-specific explanation in models of chronic disease. *Artificial Intelligence in Medicine*, **4**(3):191–205.
- M. Kajanus, J. Ahola, M. Kurttila and M. Pesonen (2001) Application of even swaps for strategy selection in a rural enterprise. *Management Decision*, **39**(5):394–402.
- R. L. Keeney and H. Raiffa (1976) *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. New York, NY: John Wiley and Sons, Inc.
- S. Lichtenstein and P. Slovic (2006) *The Construction of Preference*. Cambridge, UK: Cambridge University Press.
- J. Mustajoki and R. P. Hämäläinen (2005) A preference programming approach to make the even swaps method even easier. *Decision Analysis*, **2**(2):110–123.
- J. Mustajoki and R. P. Hämäläinen (2007) Smart-Swaps - A decision support system for multicriteria decision analysis with the even swaps method. *Decision Support Systems*, **44**:313–325.
- K. L. Poh and E. J. Horvitz (1993) Reasoning about the value of decision-model refinement: Methods and application. In *Proceedings of the 9th Conference on Uncertainty in Artificial Intelligence (UAI)*, 174–182, Morgan Kaufman.
- G. C. Scott and R. D. Shachter (2005) Individualizing generic decision models using assessments as evidence. *Journal of Biomedical Informatics*, **38**(4):281–297.
- T. Stewart (1996) Robustness of additive value function methods in MCDM. *Multi-Criteria Decision Analysis*, **5**(4):301–309.
- A. Tversky and D. Kahneman (1991) Loss aversion in riskless choice: A reference-dependent model. *The Quarterly Journal of Economics*, **106**(4):1039–1061.
- A. Tversky, S. Sattath and P. Slovic (1988) Contingent weighting in judgment and choice. *Psychological Review*, **95**(3):371–384.
- D. von Winterfeldt and W. Edwards (1986) *Decision Analysis and Behavioral Research*. Cambridge, UK: Cambridge University Press.
- M. C. Willemsen and G. Keren (2003) The meaning of indifference in choice behavior: Asymmetries in adjustments embodied in matching. *Organizational Behavior and Human Decision Processes*, **90**(2):342–359.

Learning to Predict from Crowdsourced Data

Wei Bi [†]

Liwei Wang [‡]

James T. Kwok [†]

Zhuowen Tu ^{*}

[†] Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong

[‡] Department of Computer Science, University of Illinois at Urbana-Champaign, IL, United States

^{*} Department of Cognitive Science, University of California San Diego, CA, United States

Abstract

Crowdsourcing services like Amazon’s Mechanical Turk have facilitated and greatly expedited the manual labeling process from a large number of human workers. However, spammers are often unavoidable and the crowdsourced labels can be very noisy. In this paper, we explicitly account for four sources for a noisy crowdsourced label: worker’s dedication to the task, his/her expertise, his/her default labeling judgement, and sample difficulty. A novel mixture model is employed for worker annotations, which learns a prediction model directly from samples to labels for efficient out-of-sample testing. Experiments on both simulated and real-world crowdsourced data sets show that the proposed method achieves significant improvements over the state-of-the-art.

1 INTRODUCTION

Supervised learning requires labels. However, the collection of labeled data from users is often expensive, tedious and time-consuming. Recently, the use of crowdsourcing allows this mundane process of obtaining manual labels from a great number of human workers to be greatly expedited. For example, in Amazon’s Mechanical Turk (AMT), a “requester” can pose tasks known as HITs (Human Intelligence Tasks). Workers then choose to complete any of the existing HITs and get rewarded by a certain amount of monetary payment set by the requester. Researchers in different areas, such as computer vision (Sorokin and Forsyth, 2008) and natural language processing (Snow et al., 2008), have benefited from these crowdsourcing services and acquired labels for large data sets.

However, in practice, the crowdsourced labels are often noisy. On one hand, their quality depends on the labeling task. For example, if the labeling task is not well designed or not clearly described by the requester, the worker’s motivation to participate may decrease, and the noisy level of

the crowdsourced labels will increase (Zheng et al., 2011). Moreover, different labeling tasks can have different difficulties. If samples in one task are very challenging to annotate, the obtained crowdsourced labels may be less reliable (Whitehill et al., 2009; Yan et al., 2010; Zhou et al., 2012). On the other hand, workers’ qualities can vary drastically and lead to different noise levels in their annotations. For example, their expertise differs due to their diverse knowledge backgrounds (Whitehill et al., 2009; Welinder et al., 2010). Moreover, their dedications to performing the task can also greatly affect their annotation accuracies. In the worst case, some workers may just randomly guess the labels without actually looking at the samples (Welinder et al., 2010). In particular, it is common to have “spammers”, who provide wrong labels most of the time. The extraction of “true” labels from a large pool of crowdsourced labels is thus very important.

A popular and simple approach is to perform a majority vote on workers. However, it implicitly assumes that all workers are equally accurate, which is rarely the case in practice. It can also be misleading when there is a significant portion of spammers. To obtain a more accurate consensus, a number of algorithms have been proposed that model different aspects of the labeling noise (such as worker expertise and sample difficulty) (Whitehill et al., 2009; Welinder et al., 2010; Raykar and Yu, 2012; Liu et al., 2012; Zhou et al., 2012). Interested readers are referred to the recent survey in (Sheshadri and Lease, 2013). Yet, these models can only make estimations for samples with crowdsourced labels. For out-of-sample testing (i.e. prediction on an unseen test sample), the user has to first crowdsource its labels before these algorithms can be run.

To alleviate this problem, one can build a prediction model directly from the sample to the label. A popular approach is the two-coin model (Raykar et al., 2010). It assumes that each worker generates its label by flipping the ground-truth label with a certain probability. Depending on whether the true label being zero or one, the flipping probabilities are in general different. A prediction model is then built on the hidden “denoised” labels. This is further extended in

Table 1: Comparison between the existing methods and ours.

method	prediction model from samples to labels	worker expertise	sample difficulty	worker dedication
majority voting	×	×	×	×
Whitehill et al. (2009)	×	✓	✓	×
Welinder et al. (2010)	×	✓	×	×
Liu et al. (2012)	×	✓	×	×
Zhou et al. (2012)	×	✓	✓	×
Raykar and Yu (2012)	×	✓	×	×
Raykar et al. (2010)	✓	✓	×	×
Yan et al. (2010)	✓	✓	✓	×
Kajino et al. (2012)	✓	✓	×	×
Kajino et al. (2013)	✓	✓	×	×
proposed method	✓	✓	✓	✓

(Yan et al., 2010) by allowing the flipping probability to be different from sample to sample. Another approach is to formulate the crowdsourcing problem as a multitask learning problem (Evgeniou and Pontil, 2004). Each worker is considered a task, and the final prediction model is a linear combination of the worker models (Kajino et al., 2012, 2013). However, this may not be robust when many workers are spammers or incompetent.

In this paper, we propose a novel model for the generation of crowdsourced labels. Specifically, we assume that the label noise can come from four sources: (i) the worker is not an expert; (ii) the worker is not dedicated to the task; (iii) the worker’s default label judgement is incorrect; and (iv) the sample is difficult. Note that some of these have been considered in the literature (Table 1). Moreover, they can be highly inter-correlated. For example, if a sample is easy, even an uncommitted non-expert can output the correct label. On the other side, if the sample is very difficult, even a dedicated expert can only rely on his default judgement. If his prior knowledge happens to be incorrect, the label will be wrong.

With these various factors, we employ a mixture model for the worker annotation of the crowdsourced data. If the worker is dedicated to the labeling task or if he considers the sample as easy, the corresponding label is generated according to his underlying decision function. Otherwise, the label is generated based on his default labeling judgement. To model sample difficulty, we use the usual intuition that a sample is difficult if it is close to the worker’s underlying decision boundary, and vice versa. Obviously, we do not know in which way the worker generates the label of a sample. For inference, we use the expectation maximization (EM) algorithm (Dempster et al., 1977).

The rest of this paper is organized as follows. Section 2 presents our worker annotation model, and Section 3 presents the inference procedure. Experiments are presented in Section 4, and the last section gives some concluding remarks.

2 PROPOSED MODEL

In this paper, we assume that the crowdsourced task is a binary classification problem, with T workers and N query samples. The i th sample $\mathbf{x}^{(i)} \in \mathbb{R}^d$ is annotated by the set of workers $S_i \subseteq \{1, 2, \dots, T\}$. The annotation provided by the t th worker (with $t \in S_i$) is denoted $y_t^{(i)} \in \{0, 1\}$.

2.1 GENERATION OF GROUND TRUTH

We assume that for each sample $\mathbf{x}^{(i)}$, its ground truth label $y^{*(i)} \in \{0, 1\}$ is generated by a logistic regression model with parameter \mathbf{w}^* . In other words, $y^{*(i)}$ follows the Bernoulli distribution

$$p(y^{*(i)} = 1 | \mathbf{w}^*, \mathbf{x}^{(i)}) = \sigma(\mathbf{w}^{*T} \mathbf{x}^{(i)}), \quad (1)$$

where $\sigma(z) = 1/(1 + \exp(-z))$ is the logistic function. To avoid over-fitting, we assume a normal prior on \mathbf{w}^* :

$$\mathbf{w}^* | \gamma \sim \mathcal{N}\left(\mathbf{0}, \frac{1}{\gamma} \mathbf{I}\right),$$

where $\gamma > 0$ is a constant (in the experiments, this is tuned by the validation set). Other priors can also be readily added. For example, if \mathbf{w}^* is expected to be sparse, the Laplace prior can be used instead.

As will be seen later, training the model only requires access to the features but not the ground-truth labels. This is more realistic in many crowdsourced applications, as the features can often be readily extracted using standard unsupervised feature extraction.

2.2 WORKER ANNOTATION: EXPERTISE AND DEDICATION

For worker t , we assume that his failure in correctly annotating $\mathbf{x}^{(i)}$ is due to two reasons according to his dedication to the queried sample $\mathbf{x}^{(i)}$. First, he may have tried to annotate with the best effort, but still fails because his expertise is not strong enough. We model this by assuming

that worker t 's annotation $y_t^{(i)}$ follows a similar Bernoulli distribution as (1):

$$p(y_t^{(i)} = 1 | \mathbf{w}_t, \mathbf{x}^{(i)}) = \sigma(\mathbf{w}_t^T \mathbf{x}^{(i)}), \quad (2)$$

where \mathbf{w}_t is the worker's "estimation" of \mathbf{w}^* , and is sampled from the following normal distribution

$$\mathbf{w}_t | \mathbf{w}^*, \delta_t \sim \mathcal{N}(\mathbf{w}^*, \delta_t^2 \mathbf{I}). \quad (3)$$

A small δ_t means that \mathbf{w}_t is likely to be close to \mathbf{w}^* , and thus worker t is an expert, and vice versa. When no additional information on the worker's expertise is available, a uniform hyperprior on $\{\delta_t\}_{t=1}^T$ can be used.

The second reason for worker t 's failure in correctly annotating $\mathbf{x}^{(i)}$ is simply that he is not dedicated to the task and has not even looked at $\mathbf{x}^{(i)}$. In this case, he randomly annotates according to some default judgement. This can be modeled by another Bernoulli distribution:

$$p(y_t^{(i)} = 1 | b_t) = b_t, \quad (4)$$

where $b_t \in [0, 1]$. Again, when no additional information on the worker's default labeling judgement are available, a uniform prior on $\{b_t\}_{t=1}^T$ will be used.

Combining these two causes, we have

$$p(y_t^{(i)} | \mathbf{x}^{(i)}, \mathbf{w}_t, b_t, z_t^{(i)}) = p(y_t^{(i)} | \mathbf{x}^{(i)}, \mathbf{w}_t)^{z_t^{(i)}} p(y_t^{(i)} | b_t)^{(1-z_t^{(i)})}, \quad (5)$$

where $z_t^{(i)} \in \{0, 1\}$ determines whether (3) or (4) should be used to generate $y_t^{(i)}$. Intuitively, an expert worker should have an accurate prediction model (δ_t is small), and be dedicated to the task ($z_t^{(i)} = 1$ on most $\mathbf{x}^{(i)}$'s); whereas a spammer either has a large δ_t or $z_t^{(i)} = 0$ most of the time.

2.3 INCORPORATING SAMPLE DIFFICULTY

The difficulty of a sample can greatly affect the annotation quality (Whitehill et al., 2009; Yan et al., 2010; Zhou et al., 2012). If a sample is vaguely described or too hard, even an expert may have to make a random guess and thus acts as if he has not looked at the sample. On the contrary, if a sample is very easy, even a spammer (especially the lazy ones) can quickly make a correct decision.

To model this effect on worker t , we incorporate the difficulty of $\mathbf{x}^{(i)}$ into the modeling of $z_t^{(i)}$. Intuitively, if $\mathbf{x}^{(i)}$ is difficult to annotate, $z_t^{(i)}$ should be close to 0. From (5), the annotation made is then independent of the decision model of worker t . To measure sample difficulty, we use the popular notion that worker t will perceive $\mathbf{x}^{(i)}$ as difficult if it is close to his decision boundary (Tong and Koller, 2002; Dong et al., 2013; Welinder et al., 2010). Thus, we arrive at the following Bernoulli distribution on $z_t^{(i)}$:

$$p(z_t^{(i)} = 1 | \mathbf{x}^{(i)}, \mathbf{w}_t, \lambda_t) = 2\sigma\left(\lambda_t \frac{\|\mathbf{w}_t^T \mathbf{x}^{(i)}\|^2}{\|\mathbf{w}_t\|^2}\right) - 1. \quad (6)$$

Here, $\frac{\|\mathbf{w}_t^T \mathbf{x}^{(i)}\|}{\|\mathbf{w}_t\|}$ is the distance of $\mathbf{x}^{(i)}$ from worker t 's decision boundary $\mathbf{w}_t^T \mathbf{x} = 0$, and $\lambda_t \geq 0$ models the sensitivity of worker t 's annotation to sample difficulty. Depending on each worker's expertise (as reflected by his \mathbf{w}_t), one worker may consider sample $\mathbf{x}^{(i)}$ difficult while another worker may consider it easy. Moreover, a small λ_t makes an easy sample (with a large $\frac{\|\mathbf{w}_t^T \mathbf{x}^{(i)}\|}{\|\mathbf{w}_t\|}$) look difficult and worker t will rely more on his default judgement, and vice versa. As we are only interested in the value of $\lambda_t \frac{\|\mathbf{w}_t^T \mathbf{x}^{(i)}\|^2}{\|\mathbf{w}_t\|^2}$ in (6), to simplify inference, we reparameterize (6) as

$$p(z_t^{(i)} = 1 | \mathbf{x}^{(i)}, \mathbf{w}_t, \lambda_t) = 2\sigma(\lambda_t \|\mathbf{w}_t^T \mathbf{x}^{(i)}\|^2) - 1. \quad (7)$$

After obtaining \mathbf{w}_t , the sensitivity of worker t 's annotation to sample difficulty can be recovered as $\lambda_t \|\mathbf{w}_t\|^2$.

A graphical model representation for the complete model is shown in Figure 1.

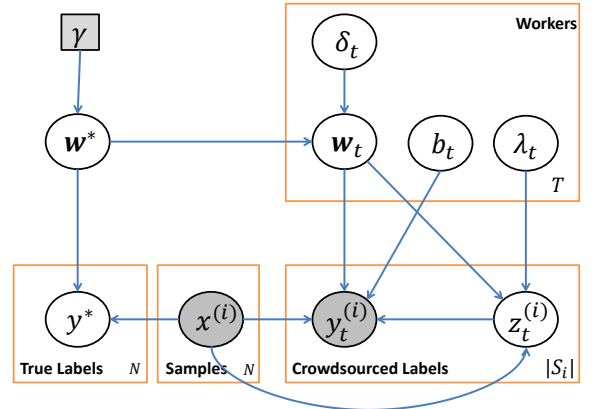


Figure 1: The proposed model incorporating sample difficulty and two sources for worker annotation.

2.4 EXTENSIONS

When the crowdsourced task is a multiclass classification problem, one can simply replace the Bernoulli distribution with a multinomial distribution. Similarly, for regression problems, the normal distribution can be used instead.

For ease of exposition, we use the linear logistic regression model in (1) and (2). This has also been used in most previous works (Raykar et al., 2010; Kajino et al., 2012). It can be easily replaced by any binary classifier. For example, to use a nonlinear kernelized version, one can replace $\mathbf{w}^{*T} \mathbf{x}^{(i)}$ in (1) by $\sum_{j=1}^N \alpha^{*(j)} k(\mathbf{x}^{(j)}, \mathbf{x}^{(i)})$, where $k(\cdot, \cdot)$ is an appropriate kernel function. Similarly, $\mathbf{w}_t^T \mathbf{x}^{(i)}$ in (2) is replaced by $\sum_{j=1}^N \alpha_t^{(j)} k(\mathbf{x}^{(j)}, \mathbf{x}^{(i)})$, where $\alpha_t = [\alpha_t^{(1)}, \dots, \alpha_t^{(N)}]$ serves as worker t 's "estimation"

of the ground truth $\alpha^* = [\alpha^{*(1)}, \dots, \alpha^{*(N)}]^T$. Analogous to (3), we can assume that each α_t is sampled from $\mathcal{N}(\alpha^*, \delta_t^2 \mathbf{I})$.

3 INFERENCE

In this section, we use the Expectation Maximization (EM) algorithm (Dempster et al., 1977) to obtain the model parameters $\Theta = \{\mathbf{w}^*, \{\mathbf{w}_t\}_{t=1}^T, \{\delta_t\}_{t=1}^T, \{b_t\}_{t=1}^T, \{\lambda_t\}_{t=1}^T\}$. Let the samples $\mathbf{X} = \{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(N)}\}$ be independent. By treating $\mathbf{Y} = \{y_t^{(i)}\}$ as the observed data and $\mathbf{Z} = \{z_t^{(i)}\}$ as the missing data, the complete data likelihood can be written as

$$\begin{aligned} \mathcal{L}(\mathbf{Y}, \mathbf{Z}) &= p(\mathbf{Y}, \mathbf{Z} | \mathbf{X}, \Theta) \\ &= p(\mathbf{Y} | \mathbf{Z}, \mathbf{X}, \{\mathbf{w}_t, b_t\}_{t=1}^T) p(\mathbf{Z} | \mathbf{X}, \{\mathbf{w}_t, \lambda_t\}_{t=1}^T) \\ &= \prod_{i=1}^N \prod_{t \in S_i} p(y_t^{(i)} | z_t^{(i)}, \mathbf{w}_t, \mathbf{x}^{(i)}, b_t) p(z_t^{(i)} | \mathbf{w}_t, \mathbf{x}^{(i)}, \lambda_t), \end{aligned} \quad (8)$$

by assuming that the workers annotate independently. The posterior of Θ is then

$$\begin{aligned} p(\mathbf{w}^*, \{\mathbf{w}_t\}_{t=1}^T, \{\delta_t\}_{t=1}^T, \{\lambda_t\}_{t=1}^T, \{b_t\}_{t=1}^T | \mathbf{X}, \mathbf{Y}, \mathbf{Z}) \\ \propto \mathcal{L}(\mathbf{Y}, \mathbf{Z}) p(\mathbf{w}^*) \prod_{t=1}^T p(\mathbf{w}_t | \mathbf{w}^*, \delta_t) p(\delta_t) p(\lambda_t) p(b_t), \end{aligned} \quad (9)$$

3.1 E-STEP

Taking the log of (8), we have

$$\begin{aligned} \log \mathcal{L}(\mathbf{Y}, \mathbf{Z}) &= \sum_{i=1}^N \sum_{t \in S_i} \left(z_t^{(i)} \log p(y_t^{(i)} | \mathbf{w}_t, \mathbf{x}^{(i)}) p(z_t^{(i)} = 1 | \mathbf{w}_t, \mathbf{x}^{(i)}, \lambda_t) \right. \\ &\quad \left. + (1 - z_t^{(i)}) \log p(y_t^{(i)} | b_t) p(z_t^{(i)} = 0 | \mathbf{w}_t, \mathbf{x}^{(i)}, \lambda_t) \right). \end{aligned}$$

The expected value of $z_t^{(i)}$, denoted $\bar{z}_t^{(i)}$, is

$$\bar{z}_t^{(i)} = \frac{1}{Q_t^{(i)}} p(y_t^{(i)} | \mathbf{w}_t, \mathbf{x}^{(i)}) p(z_t^{(i)} = 1 | \mathbf{w}_t, \mathbf{x}^{(i)}, \lambda_t),$$

where $Q_t^{(i)} = p(z_t^{(i)} = 1 | \mathbf{w}_t, \mathbf{x}^{(i)}, \lambda_t) p(y_t^{(i)} | \mathbf{w}_t, \mathbf{x}^{(i)}) + p(z_t^{(i)} = 0 | \mathbf{w}_t, \mathbf{x}^{(i)}, \lambda_t) p(y_t^{(i)} | b_t)$.

As can be seen, whether $\bar{z}_t^{(i)}$ is close to 1 is affected by both the sample difficulty (i.e., $p(z_t^{(i)} = 1 | \mathbf{w}_t, \mathbf{x}^{(i)}, \lambda_t)$) and the confidence of $y_t^{(i)}$ generated from the current estimated function \mathbf{w}_t (i.e., $p(y_t^{(i)} | \mathbf{w}_t, \mathbf{x}^{(i)})$).

¹In the kernelized version, $\Theta = \{\alpha^*, \{\alpha_t\}_{t=1}^T, \{\delta_t\}_{t=1}^T, \{b_t\}_{t=1}^T, \{\lambda_t\}_{t=1}^T\}$ and the EM procedure is similar. In particular, the M-step updates α^* and $\{\alpha_t\}_{t=1}^T$ as \mathbf{w}^* and $\{\mathbf{w}_t\}_{t=1}^T$.

3.2 M-STEP

Here, we use alternating minimization. At each step, one variable is minimized while the other variables are fixed.

- \mathbf{w}_t 's: From (9), the various \mathbf{w}_t 's can be learned independently. The optimization subproblem for \mathbf{w}_t is

$$\begin{aligned} \min_{\mathbf{w}_t} \quad & \frac{1}{\delta_t^2} \|\mathbf{w}_t - \mathbf{w}^*\|^2 - \sum_{i: t \in S_i} \left(\bar{z}_t^{(i)} y_t^{(i)} \log \sigma(\mathbf{w}_t^T \mathbf{x}^{(i)}) \right. \\ & + \bar{z}_t^{(i)} (1 - y_t^{(i)}) \log(1 - \sigma(\mathbf{w}_t^T \mathbf{x}^{(i)})) \\ & + \bar{z}_t^{(i)} \log(2\sigma(\lambda_t \|\mathbf{w}_t^T \mathbf{x}^{(i)}\|^2) - 1) \\ & \left. + (1 - \bar{z}_t^{(i)}) \log(2 - 2\sigma(\lambda_t \|\mathbf{w}_t^T \mathbf{x}^{(i)}\|^2)) \right). \end{aligned}$$

This can be maximized by gradient descent, and the gradient w.r.t. \mathbf{w}_t is

$$\begin{aligned} \frac{2}{\delta_t^2} (\mathbf{w}_t - \mathbf{w}^*) - \sum_{i: t \in S_i} \left(\bar{z}_t^{(i)} (y_t^{(i)} - \sigma(\mathbf{w}_t^T \mathbf{x}^{(i)})) \mathbf{x}^{(i)} \right. \\ \left. + \frac{(\bar{z}_t^{(i)} - 2\sigma(v_t^{(i)} + 1)\sigma(v_t^{(i)})\lambda_t \mathbf{w}_t^T \mathbf{x}^{(i)} \mathbf{x}^{(i)})}{2\sigma(v_t^{(i)}) - 1} \right), \end{aligned}$$

where $v_t^{(i)} = \lambda_t \|\mathbf{w}_t^T \mathbf{x}^{(i)}\|^2$.

- \mathbf{w}^* : The optimization subproblem for \mathbf{w}^* is

$$\min_{\mathbf{w}^*} \sum_{t=1}^T \frac{1}{\delta_t^2} \|\mathbf{w}_t - \mathbf{w}^*\|^2 + \gamma \|\mathbf{w}^*\|^2,$$

with the closed-form solution

$$\mathbf{w}^* = \frac{\sum_{t=1}^T \frac{1}{\delta_t^2} \mathbf{w}_t}{\gamma + \sum_{t=1}^T \frac{1}{\delta_t^2}}. \quad (10)$$

Note that \mathbf{w}^* is a weighted average of all the \mathbf{w}_t 's, with contributions from the experts (those with small δ_t 's) weighted heavier.

- δ_t : The optimization subproblem for δ_t is

$$\begin{aligned} \min_{\delta_t} \quad & \frac{1}{\delta_t^2} \|\mathbf{w}_t - \mathbf{w}^*\|^2 + \log \det(\delta_t^2 \mathbf{I}) \\ & = \min_{\delta_t} \frac{1}{\delta_t^2} \|\mathbf{w}_t - \mathbf{w}^*\|^2 + 2d \log \delta_t, \end{aligned}$$

where \mathbf{I} is the $d \times d$ identity matrix, and d is the number of input features. By setting its derivative w.r.t. δ_t to 0, we obtain

$$\delta_t = \frac{1}{\sqrt{d}} \|\mathbf{w}_t - \mathbf{w}^*\|.$$

- b_t : The optimization subproblem is

$$\max_{b_t} \sum_{i: t \in S_i} (1 - \bar{z}_t^{(i)}) (y_t^{(i)} \log b_t + (1 - y_t^{(i)}) \log(1 - b_t)).$$

By setting its derivative w.r.t. b_t to 0, we have

$$\sum_{i:t \in S_i} (1 - \bar{z}_t^{(i)}) \left(\frac{y_t^{(i)}}{b_t} - \frac{1 - y_t^{(i)}}{1 - b_t} \right) = 0.$$

Rearranging gives the closed-form solution

$$b_t = \frac{\sum_{i:t \in S_i} (1 - \bar{z}_t^{(i)}) y_t^{(i)}}{\sum_{i:t \in S_i} (1 - \bar{z}_t^{(i)})}.$$

Recall that $\bar{z}_t^{(i)} \in [0, 1]$ is the expectation of $y_t^{(i)}$ generated by worker t 's default judgement. Hence, b_t is simply the average of worker t 's labels that are generated by his default judgement.

- $\{\lambda_t\}_{t=1}^T$: The optimization subproblem is

$$\begin{aligned} \max_{\lambda_t} \sum_{i:t \in S_i} & \bar{z}_t^{(i)} \log(2\sigma(\lambda_t \|\mathbf{w}_t^T \mathbf{x}^{(i)}\|^2) - 1) \\ & + (1 - \bar{z}_t^{(i)}) \log(2 - 2\sigma(\lambda_t \|\mathbf{w}_t^T \mathbf{x}^{(i)}\|^2)). \end{aligned}$$

Again, this can be solved by projected gradient (as $\lambda_t \geq 0$), with the gradient w.r.t. λ_t given by

$$\sum_{i:t \in S_i} \frac{(\bar{z}_t^{(i)} - 2\sigma(v_t^{(i)}) + 1)\sigma(v_t^{(i)}) \|\mathbf{w}_t^T \mathbf{x}^{(i)}\|^2}{2\sigma(v_t^{(i)}) - 1}.$$

4 EXPERIMENTS

In this section, we perform two sets of experiments to evaluate the performance of the proposed method. Section 4.2 simulates a crowdsourced environment with synthetic workers using a standard benchmark data set; while Section 4.3 uses data sets with real labels crowdsourced from the AMT.

4.1 SETUP

The proposed model will be compared with the following groups of algorithms:

1. Algorithms that learn prediction models directly from samples to labels (Table 1). In particular, we will compare with
 - MTL: The multitask formulation in (Kajino et al., 2012). Each worker is considered as a task, and the prediction model is a rescaled average of all the learned worker models.
 - RY: The two-coin model in (Raykar et al., 2010). It considers the annotation generated by flipping the ground truth label with a certain biased probability.

- YAN: This model is proposed in (Yan et al., 2010), and an extension of (Raykar et al., 2010). Its flipping probability is sample-specific and varies with sample difficulty. However, unlike ours, it does not have a clear connection with the worker's decision function.

2. Algorithms that do not learn a prediction model from samples to labels (Table 1). In particular, we will compare with

- GLAD (Whitehill et al., 2009)²: It models each sample's difficulty level and each worker's expertise.
- CUBAM (Welinder et al., 2010)³:: It considers sample competence, worker expertise and bias.
- MV: Majority voting, a popular baseline which essentially treats all the workers as equally accurate.

For prediction on an unseen test sample, these algorithms have to first crowdsource its labels. To avoid this problem, we will proceed as follows: (i) Estimate the "true" labels of the training samples using each of these algorithms; (ii) Use the estimated labels to train a logistic regression model; (iii) Use the trained regression model to make predictions on the test samples.

3. We also include an ideal baseline (Ideal), which is a logistic regression model trained using the training samples with ground truth labels.

For performance evaluation, we follow (Raykar et al., 2010) and report the area under ROC curve (AUC). The ROC curve is obtained by varying the prediction threshold. Parameters in all the models are tuned by a validation set (which is constructed by using 20% of the training data). With the chosen parameters, a prediction model is then learned using all the training data.

4.2 UCI DATA SET

Following (Kajino et al., 2012), we use the red wine data in the UCI Wine-Quality data set⁴. There are a total of 1,599 samples, each with 11 features. The original multiclass labels are binarized such that samples with quality levels below 7 are labeled as 0, and 1 otherwise. 70% of the samples are randomly chosen for training, and the remaining 30% for testing. To reduce statistical variability, results are averaged over 5 repetitions.

²Code is from <http://mplab.ucsd.edu/~jake/>

³Code is from <http://www.vision.caltech.edu/welinder/cubam.html>

⁴<http://archive.ics.uci.edu/ml/datasets/Wine+Quality>

Table 2: Testing AUCs on the wine data set. The best results and those that are not statistically worse (according to the paired t-test with p-value less than 0.05) are in bold.

	#workers	proposed	MTL	RY	YAN	GLAD	CUBAM	MV	Ideal
set 1	20	0.81 \pm 0.01	0.79 \pm 0.04	0.38 \pm 0.04	0.49 \pm 0.03	0.52 \pm 0.02	0.66 \pm 0.05	0.48 \pm 0.06	0.87 \pm 0.02
	40	0.79 \pm 0.01	0.75 \pm 0.07	0.51 \pm 0.01	0.53 \pm 0.03	0.51 \pm 0.03	0.58 \pm 0.04	0.34 \pm 0.02	0.87 \pm 0.03
set 2	20	0.81 \pm 0.01	0.80 \pm 0.01	0.50 \pm 0.03	0.49 \pm 0.03	0.49 \pm 0.05	0.52 \pm 0.04	0.49 \pm 0.01	0.87 \pm 0.01
	40	0.73 \pm 0.04	0.76 \pm 0.02	0.49 \pm 0.01	0.50 \pm 0.03	0.50 \pm 0.03	0.54 \pm 0.03	0.50 \pm 0.03	0.79 \pm 0.02
set 3	20	0.80 \pm 0.01	0.82 \pm 0.01	0.80 \pm 0.03	0.78 \pm 0.03	0.84 \pm 0.05	0.84 \pm 0.04	0.48 \pm 0.03	0.84 \pm 0.01
	40	0.80 \pm 0.04	0.82 \pm 0.02	0.80 \pm 0.01	0.76 \pm 0.04	0.84 \pm 0.05	0.84 \pm 0.03	0.59 \pm 0.03	0.85 \pm 0.02

4.2.1 Generation of Labels

We generate three sets of simulated crowdsourced labels based on different model assumptions:

- Set 1: The crowdsourced labels are generated using the proposed annotation process. The “optimal” \mathbf{w}^* is obtained by training a logistic regression model on all the training and test samples. We generate different numbers (20 and 40) of noisy workers. For each worker, we generate \mathbf{w}_t as in (3) with different settings of δ_t ’s:
 1. $\frac{1}{4}$ of the workers have $\delta_t = 10$ (high expertise);
 2. $\frac{1}{2}$ of the workers have $\delta_t = 100$ (moderate expertise); and
 3. $\frac{1}{4}$ of the workers have $\delta_t = 1000$ (low expertise).

Sample difficulty is generated as in Section 2.3:

1. For the expert workers, we set $\lambda_t = 10,000$, and so most of the samples appear easy;
2. For workers with moderate expertise, set $\lambda_t = 100$; and
3. For workers with low expertise, set $\lambda_t = 1$ (and so most of the samples appear difficult).

For each sample i , we set $z_t^{(i)} = 1$ with probability given in (7). If $z_t^{(i)} = 1$, $y_t^{(i)}$ is labeled 1 with probability defined in (2); otherwise, $y_t^{(i)}$ is always labeled 1 (i.e., b_t in (4) is set to 1).

In summary, $\frac{1}{4}$ of the workers are experts, $\frac{1}{2}$ of them are non-experts but not very noisy; while the remaining $\frac{1}{4}$ are very noisy workers.

- Set 2: The crowdsourced labels are generated using the MTL assumption in (Kajino et al., 2012). Specifically, from the \mathbf{w}_t generated in Set 1, we generate $y_t^{(i)} = 1$ with probability $\sigma(\mathbf{w}_t^T \mathbf{x}^{(i)})$.
- Set 3: The crowdsourced labels are generated using the two-coin assumption in (Raykar et al., 2010). For worker t , let α_t (resp. β_t) be the probability that a ground truth label with value 1 (resp. 0) is flipped.

1. For $\frac{1}{4}$ of the workers, we set $\alpha_t = \beta_t = 0.05$ (experts);
2. For $\frac{1}{2}$ of the workers, set $\alpha_t = \beta_t = 0.25$ (non-experts but not very noisy); and
3. For $\frac{1}{4}$ of the workers, set $\alpha_t = \beta_t = 0.55$ (very noisy workers).

4.2.2 Results on ROC Curves

Figure 2 shows the obtained testing ROC curves (with each point averaged over the five repetitions). The corresponding averaged AUC values are shown in Table 2. As can be seen, the proposed model performs well under various noise generation scenarios.

On Set 1, since the labels are generated using the proposed annotation process, the proposed method performs significantly better than the others as expected. MTL is the second best, as it also builds a prediction model for each worker. RY, YAN, GLAD, CUBAM and MV perform poorly, as their model assumptions are very different from the underlying data generation process.

On Set 2, MTL is the best. The proposed model also yields comparable performance; while the others do not perform well.

On Set 3, MTL, RY, GLAD, CUBAM and the proposed method have comparable performance. Their performance gaps with Ideal are also quite small, which is consistent with the results in (Kajino et al., 2012). As various methods can perform well here, it suggests that the noise generated by the two-coin model is easier to remove than those in the previous two settings.

4.2.3 Separating Experts from Noisy Workers

In this section, we examine the proposed model’s ability to separate experts from noisy workers using the two criteria: worker expertise and worker dedication. Because of the lack of space, we will only show results (averaged over the 5 repetitions) on Set 1.

First, we check if the proposed model can detect workers with high expertise. Figures 3(a) and 3(b) show the contri-

bution of \mathbf{w}_t in \mathbf{w}^* in (10) (i.e., $\frac{1}{\delta_t^2}/(\gamma + \sum_j \frac{1}{\delta_j^2})$). As can be seen, all the nonzero contributions are from the experts, while the other workers are barely used.

Next, we check if the proposed model can find the dedicated workers. Recall that for experts, most of his $z_t^{(i)}$'s should be close to 1; while most of the $z_t^{(i)}$'s for non-dedicated workers are close to 0. Figures 3(c) and 3(d) show the value of $\hat{z}_t = \sum_{i:t \in S_i} \bar{z}_t^{(i)}$ for each worker. As expected, the \hat{z} 's of experts are large; while those of the others are usually much smaller (especially for the noisy workers).

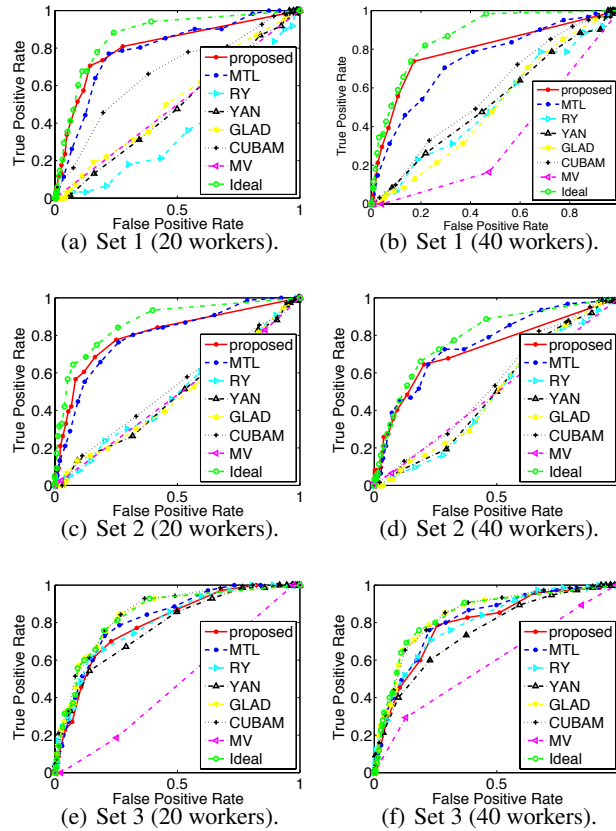


Figure 2: Testing ROC curves on the wine data set.

4.3 DATA SETS CROWDSOURCED FROM AMT

4.3.1 Data Collection and Feature Extraction

For better performance evaluation, it is desirable for the data set to satisfy the following three conditions: (i) It is labeled by a sufficient number of workers so that workers with different expertise and dedications are all involved; (ii) Each worker labels a sufficient amount of data so that one can reliably model the annotating behavior of each worker; (iii) The ground truth labels are provided. To our best knowledge, very few crowdsourced data sets meet

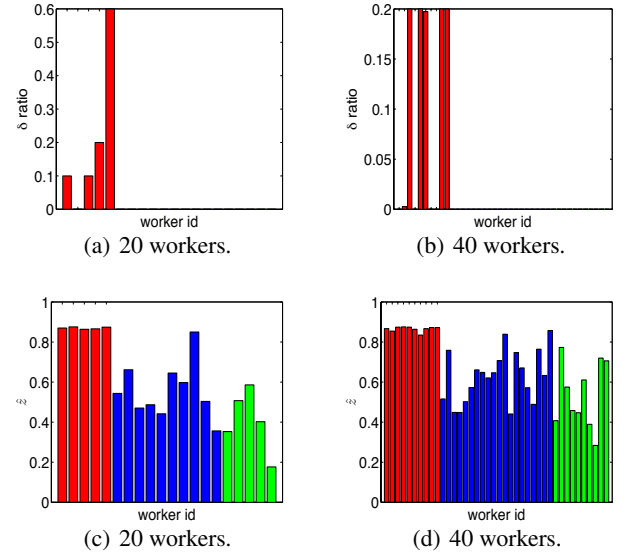


Figure 3: Worker expertise and worker dedication on the Set 1 data. 3(a) and 3(b): Contribution of each worker t towards \mathbf{w}^* ($\frac{1}{\delta_t^2}/(\gamma + \sum_j \frac{1}{\delta_j^2})$). 3(c) and 3(d): Average \hat{z} 's of the workers. Columns in red/blue/green correspond to experts/non-experts/noisy workers. In 3(a) and 3(c): workers 1-5 are experts; 6-15 are non-experts; and 16-20 are noisy workers. In 3(b) and 3(d): workers 1-10 are experts; 11-30 are non-experts; and 31-40 are noisy workers.

all these requirements. Thus, in the following, we build a crowdsourced data set based on the Stanford Dog data set⁵ (Khosla et al., 2011). It contains images of 120 breeds (categories) of dogs collected from the ImageNet⁶ (Deng et al., 2009).

For an image, its raw pixel representation is very high-dimensional and also sensitive to image changes such as scales, object locations, illuminations. Consequently, various image features have been studied by the computer vision community to better represent the image from low level (e.g. SIFT (Lowe, 1999)) to mid-level descriptors (Wang et al., 2012). In this experiment, we extract 4,096-dimensional features from images using the DeCAF (deep convolutional activation feature) algorithm (Donahue et al., 2014). These features are outputs from the intermediate layers of a pre-trained deep convolutional neural network (Krizhevsky et al., 2012). It has been shown that they can be used as generic representations for various vision tasks, and have achieved good performance even when combined with simple linear classifiers (Donahue et al., 2014).

⁵<http://vision.stanford.edu/aditya86/ImageNetDogs/>

⁶<http://www.image-net.org/>



Figure 4: Sample images of the 10 dog categories.

4.3.2 Setup

We select the 10 categories that are most difficult to classify (Khosla et al., 2011) (Figure 4). For each category, images belonging to this category are taken as positive samples; while images from the other categories are treated as negative samples. Some statistics of the data sets are shown in Table 3. The constructed data sets are then randomly split into HITs on the AMT. Each HIT contains 50 images and is labeled by 6 workers. There are a total of 65 HITS and 21 workers over the 10 categories.

Table 3: Statistics on the dog data sets.

data set	#positive sample	#negative sample	avg #samples per worker
Chihuahua	142	157	85
Japanese spaniel	142	157	85
Maltese dog	142	163	85
Pekinese	142	163	85
Shih-Tzu	142	157	83
Blenheim spaniel	142	207	89
Papillon	142	175	92
Toy terrier	142	175	86
Rhodesian ridgeback	142	207	88
Afghan hound	142	207	89

For each category, we randomly use 50% of the samples for training, and the rest for testing. To reduce statistical variability, results are averaged over 5 repetitions.

4.3.3 Results on ROC Curves

The ROC curves are shown in Figure 5, and the corresponding AUC values in Table 4. As can be seen, the proposed method yields the highest AUC on all 10 categories. It is then followed by CUBAM, GLAD, RY and YAN, which are very competitive on some categories. MTL can sometimes achieve good performance (e.g., Blenheim spaniel), but are often much inferior. Overall, the simple MV is the worst.

4.3.4 Experts vs Noisy Workers

As in Section 4.2.3, we examine the obtained δ_t 's and \bar{z}_t 's on the Chihuahua, Japanese spaniel and Maltese dog categories. As the real experts and noisy workers are not known, we assume that workers with high overall accuracy

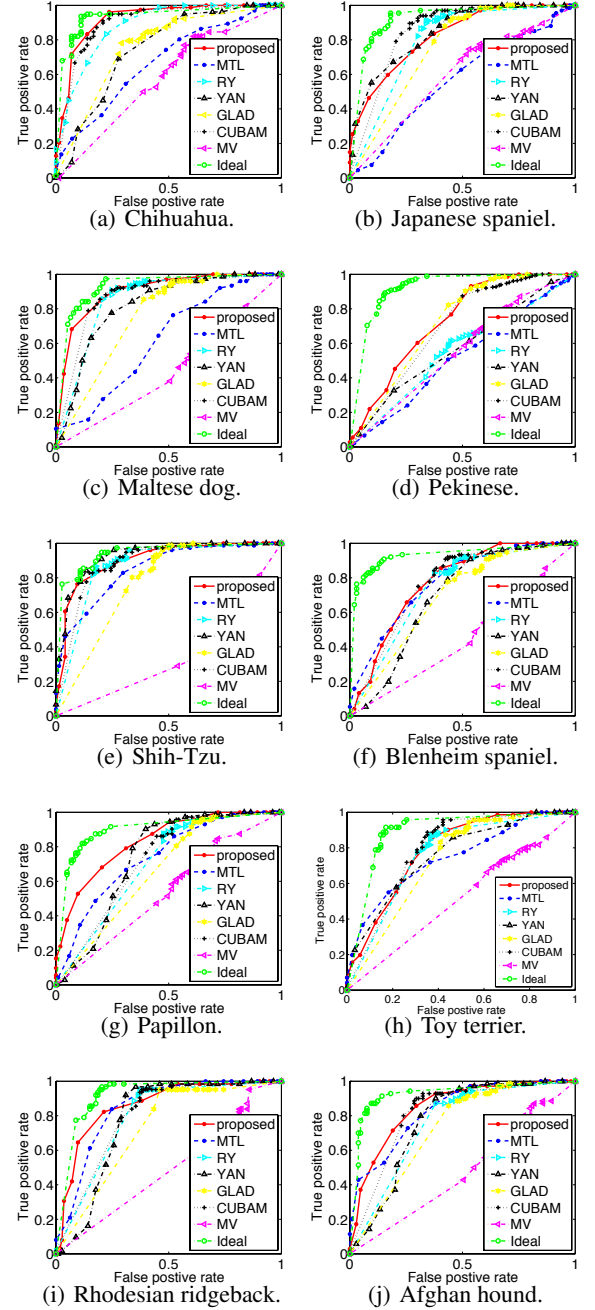


Figure 5: Testing ROC curves of the dog data sets.

Table 4: Testing AUCs on the **dog** data sets. The best results and those that are not statistically worse (according to the paired t-test with p-value less than 0.05) are in bold.

data set	proposed	MTL	RY	YAN	GLAD	CUBAM	MV	Ideal
Chihuahua	0.92 \pm 0.02	0.67 \pm 0.01	0.88 \pm 0.04	0.74 \pm 0.08	0.76 \pm 0.06	0.90 \pm 0.02	0.58 \pm 0.11	0.94 \pm 0.02
Japanese spaniel	0.83 \pm 0.01	0.57 \pm 0.01	0.80 \pm 0.03	0.84 \pm 0.04	0.75 \pm 0.04	0.85 \pm 0.03	0.60 \pm 0.05	0.92 \pm 0.01
Maltese dog	0.90 \pm 0.01	0.62 \pm 0.05	0.85 \pm 0.02	0.82 \pm 0.03	0.76 \pm 0.05	0.87 \pm 0.03	0.43 \pm 0.02	0.93 \pm 0.02
Pekinese	0.73 \pm 0.05	0.53 \pm 0.02	0.60 \pm 0.03	0.58 \pm 0.04	0.72 \pm 0.05	0.72 \pm 0.03	0.56 \pm 0.09	0.92 \pm 0.01
Shih-Tzu	0.90 \pm 0.02	0.85 \pm 0.03	0.87 \pm 0.04	0.93 \pm 0.03	0.77 \pm 0.03	0.88 \pm 0.03	0.35 \pm 0.08	0.94 \pm 0.03
Blenheim spaniel	0.78 \pm 0.03	0.78 \pm 0.03	0.74 \pm 0.02	0.69 \pm 0.05	0.69 \pm 0.07	0.77 \pm 0.03	0.45 \pm 0.03	0.93 \pm 0.03
Papillon	0.83 \pm 0.03	0.74 \pm 0.07	0.70 \pm 0.05	0.74 \pm 0.04	0.66 \pm 0.04	0.72 \pm 0.04	0.53 \pm 0.06	0.90 \pm 0.03
Toy terrier	0.79 \pm 0.02	0.75 \pm 0.01	0.76 \pm 0.03	0.76 \pm 0.03	0.73 \pm 0.02	0.79 \pm 0.04	0.51 \pm 0.05	0.89 \pm 0.03
Rhodesian ridgeback	0.86 \pm 0.04	0.85 \pm 0.03	0.79 \pm 0.02	0.78 \pm 0.02	0.73 \pm 0.05	0.79 \pm 0.04	0.50 \pm 0.05	0.92 \pm 0.01
Afghan hound	0.85 \pm 0.02	0.83 \pm 0.02	0.76 \pm 0.01	0.77 \pm 0.01	0.73 \pm 0.04	0.81 \pm 0.04	0.47 \pm 0.05	0.93 \pm 0.01

cies (that are computed based on both the training and test samples) are experts. In Figures 6(a),(c) and (e), we first plot the overall accuracies versus average weighting of the workers ($\frac{1}{\delta_t^2}/(\gamma + \sum_j \frac{1}{\delta_j^2})$) over five repetitions. As can be seen, workers with high weights, which are detected as experts in our model, generally have high overall accuracies. Next, we plot the overall accuracies versus average \hat{z}_t 's of the workers over five repetitions (Figures 6(b),(d) and (e)). Workers with high average \hat{z}_t 's are detected as dedicated workers and those with low average \hat{z}_t 's as lazy workers. As shown, the detected dedicated workers generally have high overall accuracies.

5 CONCLUSION

In this paper, we proposed a new model for crowdsourced labels that can perform out-of-sample prediction effectively. We observe that the worker's expertise and dedication to the task greatly affect the labeling process. We employed a mixture of distributions to model the annotation process: one models the worker's expertise and the other one depicts worker's labeling judgement with his random guess. We showed that this model can be easily extended to account for sample difficulty. The proposed model can be solved by the simple EM algorithm. Experiments on both UCI and real-world crowdsourced data sets demonstrate that the proposed method has significant improvements over other state-of-the-art approaches.

Acknowledgements

This research was supported in part by the Research Grants of the Hong Kong Special Administrative Region (Grant 614513), IIS-1360566 (NSF IIS-1216528) and NSF IIS-1360568 (IIS-0844566). Also, we thank Jiajun Wu for helping collecting part of the AMT data sets.

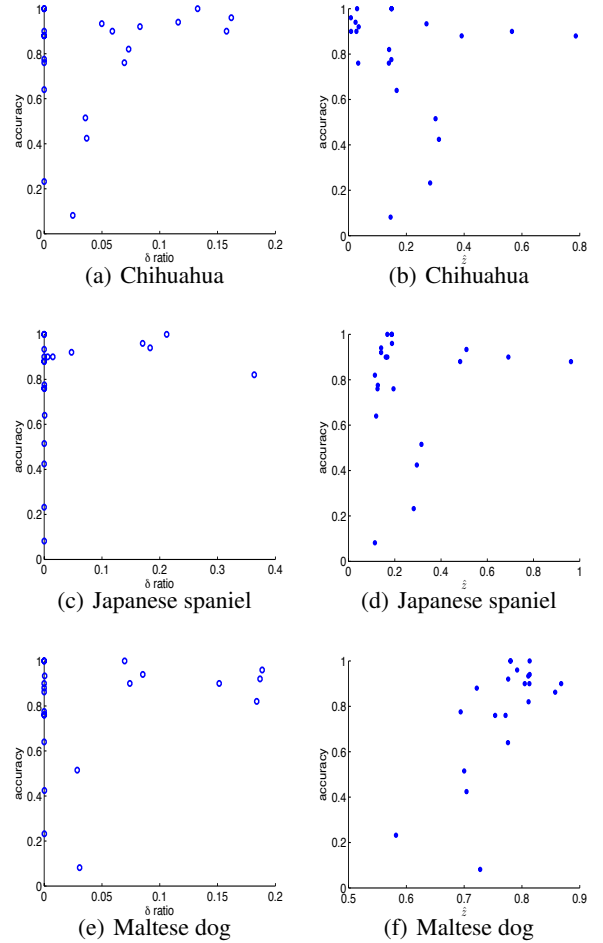


Figure 6: Results on the Chihuahua, Japanese spaniel and Maltese dog data sets. Figures 6(a), 6(c) and 6(e): Overall accuracies vs average weighting of the workers ($\frac{1}{\delta_t^2}/(\gamma + \sum_j \frac{1}{\delta_j^2})$); Figures 6(b), 6(d) and 6(f): Overall accuracies vs average $\hat{z}_t^{(i)}$'s of all workers.

References

- A. P. Dempster, N. M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B*, pages 1–38, 1977.
- J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, Miami, FL, USA, 2009.
- J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, Er. Tzeng, and T. Darrell. DeCAF: A deep convolutional activation feature for generic visual recognition. In *Proceedings of the International Conference on Machine Learning*, pages 647–655, Beijing, China, 2014.
- J. Dong, W. Xia, Q. Chen, J. Feng, Z. Huang, and S. Yan. Subcategory-aware object classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 827–834, Portland, OR, USA, June 2013.
- T. Evgeniou and M. Pontil. Regularized multi-task learning. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pages 109–117, Seattle, WA, USA, 2004.
- H. Kajino, Y. Tsuboi, and H. Kashima. A convex formulation for learning from crowds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Toronto, Canada, 2012.
- H. Kajino, Y. Tsuboi, and H. Kashima. Clustering crowds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Bellevue, WA, USA, 2013.
- A. Khosla, N. Jayadevaprakash, B. Yao, and L. Fei-Fei. Novel dataset for fine-grained image categorization. In *the Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*, Colorado Springs, CO, USA, 2011.
- A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1106–1114, Lake Tahoe, NV, USA, 2012.
- Q. Liu, J. Peng, and A. Ihler. Variational inference for crowdsourcing. In *Advances in Neural Information Processing Systems 25*, pages 701–709, Lake Tahoe, NV, USA, 2012.
- D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1150–1157, Ft. Collins, CO, USA, 1999.
- V. C. Raykar and S. Yu. Eliminating spammers and ranking annotators for crowdsourced labeling tasks. *Journal of Machine Learning Research*, 13:491–518, 2012.
- V. C. Raykar, S. Yu, L. H. Zhao, G. H. Valadez, C. Florin, L. Bogoni, and L. Moy. Learning from crowds. *Journal of Machine Learning Research*, 99:1297–1322, 2010.
- A. Sheshadri and M. Lease. SQUARE: A benchmark for research on computing crowd consensus. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, pages 156–164, Palm Springs, CA, USA, 2013.
- R. Snow, B. O’Connor, D. Jurafsky, and A. Y. Ng. Cheap and fast – but is it good?: Evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 254–263, Honolulu, Hawaii, 2008.
- A. Sorokin and D. Forsyth. Utility data annotation with amazon mechanical turk. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8, Anchorage, AK, USA, 2008.
- S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66, 2002.
- L. Wang, Y. Li, J. Jia, J. Sun, D. Wipf, and J. M. Rehg. Learning sparse covariance patterns for natural scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2767–2774, Providence, RI, USA, 2012.
- P. Welinder, S. Branson, P. Perona, and S. J. Belongie. The multidimensional wisdom of crowds. In *Advances in Neural Information Processing Systems 22*, pages 2424–2432, Vancouver, Canada, 2010.
- J. Whitehill, T.-F. Wu, J. Bergsma, J. R. Movellan, and P. L. Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in Neural Information Processing Systems 22*, pages 2035–2043, Vancouver, Canada, 2009.
- Y. Yan, R. Rosales, G. Fung, M. W. Schmidt, G. H. Valadez, L. Bogoni, L. Moy, and J. G. Dy. Modeling annotator expertise: Learning when everybody knows a bit of something. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pages 932–939, Chia Laguna, Italy, 2010.
- H. Zheng, D. Li, and W. Hou. Task design, motivation, and participation in crowdsourcing contests. *International Journal of Electronic Commerce*, 15(4):57–88, 2011.
- D. Zhou, J. C. Platt, S. Basu, and Y. Mao. Learning from the wisdom of crowds by minimax entropy. In *Advances in Neural Information Processing Systems 25*, pages 2204–2212, Lake Tahoe, NV, USA, 2012.

Lifted Tree-Reweighted Variational Inference

Hung Hai Bui

Natural Language Understanding Lab
Nuance Communications
Sunnyvale, CA, USA
bui.h.hung@gmail.com

Tuyen N. Huynh

John von Neumann Institute
Vietnam National University
Ho Chi Minh City
tuyen.huynh@jvn.edu.vn

David Sontag

Courant Institute of Mathematical Sciences
New York University
dsontag@cs.nyu.edu

Abstract

We analyze variational inference for highly symmetric graphical models such as those arising from first-order probabilistic models. We first show that for these graphical models, the tree-reweighted variational objective lends itself to a compact lifted formulation which can be solved much more efficiently than the standard TRW formulation for the ground graphical model. Compared to earlier work on lifted belief propagation, our formulation leads to a convex optimization problem for lifted marginal inference and provides an upper bound on the partition function. We provide two approaches for improving the lifted TRW upper bound. The first is a method for efficiently computing maximum spanning trees in highly symmetric graphs, which can be used to optimize the TRW edge appearance probabilities. The second is a method for tightening the relaxation of the marginal polytope using lifted cycle inequalities and novel exchangeable cluster consistency constraints.

1 Introduction

Lifted probabilistic inference focuses on exploiting symmetries in probabilistic models for efficient inference [5, 2, 3, 10, 17, 18, 21]. Work in this area has demonstrated the possibility to perform very efficient inference in highly-connected, large tree-width, but *symmetric* models, such as those arising in the context of relational (first-order) probabilistic models and exponential family random graphs [19]. These models also arise frequently in probabilistic programming languages, an area of increasing importance as demonstrated by DARPA’s PPAML program (Probabilistic Programming for Advancing Machine Learning).

Even though lifted inference can sometimes offer order-of-magnitude improvement in performance, approximation is still necessary. A topic of particular interest is the interplay between lifted inference and variational approximate infer-

ence. Lifted loopy belief propagation (LBP) [13, 21] was one of the first attempts at exploiting symmetry to speed up loopy belief propagation; subsequently, counting belief propagation (CBP) [16] provided additional insights into the nature of symmetry in BP. Nevertheless, these work were largely procedural and specific to the choice of message-passing algorithm (in this case, loopy BP). More recently, Bui et al., [3] proposed a general framework for lifting a broad class of convex variational techniques by formalizing the notion of symmetry (defined via automorphism groups) of graphical models and the corresponding variational optimization problems themselves, independent of any specific methods or solvers.

Our goal in this paper is to extend the lifted variational framework in [3] to address the important case of approximate marginal inference. In particular, we show how to lift the tree-reweighted (TRW) convex formulation of marginal inference [28]. As far as we know, our work presents the first lifted *convex* variational marginal inference, with the following benefits over previous work: (1) a lifted convex upper bound of the log-partition function, (2) a new tightening of the relaxation of the lifted marginal polytope exploiting exchangeability, and (3) a convergent inference algorithm. We note that convex upper bounds of the log-partition function immediately lead to concave lower bounds of the log-likelihood which can serve as useful surrogate loss functions in learning and parameter estimation [29, 13].

To achieve the above goal, we first analyze the symmetry of the TRW log-partition and entropy bounds. Since TRW bounds depend on the choice of the edge appearance probabilities ρ , we prove that the quality of the TRW bound is not affected if one only works with suitably symmetric ρ . Working with symmetric ρ gives rise to an explicit lifted formulation of the TRW optimization problem that is equivalent but much more compact. This convex objective function can be convergently optimized via a Frank-Wolfe (conditional gradient) method where each Frank-Wolfe iteration solves a lifted MAP inference problem. We then discuss the optimization of the edge-appearance vector ρ , effectively yielding a lifted algorithm for computing maxi-

mum spanning trees in symmetric graphs.

As in Bui et al.’s framework, our work can benefit from any tightening of the local polytope such as the use of cycle inequalities [1, 23]. In fact, each method for relaxing the marginal polytope immediately yields a variant of our algorithm. Notably, in the case of exchangeable random variables, radically sharper tightening (sometimes even exact characterization of the lifted marginal polytope) can be obtained via a set of simple and elegant linear constraints which we call *exchangeable polytope constraints*. We provide extensive simulation studies comparing the behaviors of different variants of our algorithm with exact inference (when available) and lifted LBP demonstrating the advantages of our approach. The supplementary material [4] provides additional proof and algorithm details.

2 Background

We begin by reviewing variational inference and the tree-reweighted (TRW) approximation. We focus on inference in Markov random fields, which are distributions in the exponential family given by $\Pr(x; \theta) = \exp \{ \langle \Phi(x), \theta \rangle - A(\theta) \}$, where $A(\theta)$ is called the *log-partition function* and serves to normalize the distribution. We assume that the random variables $x \in \mathcal{X}^n$ are discrete-valued, and that the features (Φ_i) , $i \in \mathcal{I}$ factor according to the graphical model structure \mathcal{G} ; Φ can be non-pairwise and is assumed to be overcomplete. This paper focuses on the inference tasks of estimating the marginal probabilities $p(x_i)$ and approximating the log-partition function. Throughout the paper, the domain \mathcal{X} is the binary domain $\{0, 1\}$, however, except for the construction of exchangeable polytope constraints in Section 6, this restriction is not essential.

Variational inference approaches view the log-partition function as a convex optimization problem over the marginal polytope $A(\theta) = \sup_{\mu \in \mathcal{M}(\mathcal{G})} \langle \mu, \theta \rangle - A^*(\mu)$ and seek tractable approximations of the negative entropy A^* and the marginal polytope \mathcal{M} [27]. Formally, $-A^*(\mu)$ is the entropy of the maximum entropy distribution with moments μ . Observe that $-A^*(\mu)$ is upper bounded by the entropy of the maximum entropy distribution consistent with any subset of the expected sufficient statistics μ . To arrive at the TRW approximation [26], one uses a subset given by the pairwise moments of a spanning tree¹. Hence for any distribution ρ over spanning trees, an upper bound on $-A^*$ is obtained by taking a convex combination of tree entropies $-B^*(\tau, \rho) = \sum_{s \in V(\mathcal{G})} H(\tau_s) - \sum_{e \in E(\mathcal{G})} I(\tau_e) \rho_e$. Since ρ is a distribution over spanning trees, it must belong to the spanning tree polytope $\mathbb{T}(\mathcal{G})$ with ρ_e denoting the edge appearance probability of e . Combined with a relaxation of the marginal polytope $\text{OUTER} \supset \mathcal{M}$, an upper

bound B of the log-partition function is obtained:

$$A(\theta) \leq B(\theta, \rho) = \sup_{\tau \in \text{OUTER}(\mathcal{G})} \langle \tau, \theta \rangle - B^*(\tau, \rho) \quad (1)$$

We note that B^* is linear w.r.t. ρ , and for $\rho \in \mathbb{T}(\mathcal{G})$, B^* is convex w.r.t. τ . On the other hand, B is convex w.r.t. ρ and θ .

The optimal solution $\tau^*(\rho, \theta)$ of the optimization problem (1) can be used as an approximation to the mean parameter $\mu(\theta)$. Typically, the local polytope LOCAL given by pairwise consistency constraints is used as the relaxation OUTER; in this paper, we also consider tightening of the local polytope.

Since (1) holds with any edge appearance ρ in the spanning tree polytope \mathbb{T} , the TRW bound can be further improved by optimizing ρ

$$\inf_{\rho \in \mathbb{T}(\mathcal{G})} B(\theta, \rho) \quad (2)$$

The resulting ρ^* is then plugged into (1) to find the marginal approximation. In practice, one might choose to work with some fixed choice of ρ , for example the uniform distribution over all spanning trees. [14] proposed using the most uniform edge-weight $\arg \inf_{\rho \in \mathbb{T}(\mathcal{G})} \sum_{e \in E} (\rho_e - \frac{|V|-1}{|E|})^2$ which can be found via conditional gradient where each direction-finding step solves a maximum spanning tree problem.

Several algorithms have been proposed for optimizing the TRW objective (1) given fixed edge appearance probabilities. [27] derived the tree-reweighted belief propagation algorithm from the fixed point conditions. [8] show how to solve the dual of the TRW objective, which is a geometric program. Although this algorithm has the advantage of guaranteed convergence, it is non-trivial to generalize this approach to use tighter relaxations of the marginal polytope, which we show is essential for lifted inference. [14] use an explicit set of spanning trees and then use dual decomposition to solve the optimization problem. However, as we show in the next section, to maintain symmetry it is essential that one *not* work directly with spanning trees but rather use symmetric edge appearance probabilities. [23] optimize TRW over the local and cycle polytopes using a Frank-Wolfe (conditional gradient) method, where each iteration requires solving a linear program. We follow this latter approach in our paper.

To optimize the edge appearance in (2), [26] proposed using conditional gradient. They observed that $\frac{\partial B(\theta, \rho)}{\partial \rho_e} = -\frac{\partial B^*(\tau^*, \rho)}{\partial \rho_e} = -I(\tau_e^*)$ where τ^* is the solution of (1). The direction-finding step in conditional gradient reduces to solving $\sup_{\rho \in \mathbb{T}(\mathcal{G})} \langle \rho, I \rangle$, again equivalent to finding the maximum spanning tree with edge mutual information $I(\tau_e^*)$ as weights. We discuss the corresponding lifted problem in section 5.

¹If the original model contains non-pairwise potentials, they can be represented as cliques in the graphical model, and the bound based on spanning trees still holds.

3 Lifted Variational Framework

We build on the key element of the lifted variational framework introduced in [3]. The automorphism group of a graphical model, or more generally, an exponential family is defined as the group \mathbb{A} of permutation pairs (π, γ) where π permutes the set of variables and γ permutes the set of features in such a way that they preserve the feature function: $\Phi^{\gamma^{-1}}(x^\pi) = \Phi(x)$. Note that this construction of \mathbb{A} is entirely based on the structure of the model and does not depend on the particular choice of the model parameters; nevertheless the group stabilizes² (preserves) the key characteristics of the exponential family such as the marginal polytope \mathcal{M} , the log-partition A and entropy $-A^*$.

As shown in [3] the automorphism group is particularly useful for exploiting symmetries when parameters are tied. For a given parameter-tying partition Δ such that $\theta_i = \theta_j$ for i, j in the same cell³ of Δ , the group \mathbb{A} gives rise to a subgroup called the lifting group \mathbb{A}_Δ that stabilizes the tied-parameter vector θ as well as the exponential family. The orbit partition of the the lifting group can be used to formulate equivalent but more compact variational problems. More specifically, let $\varphi = \varphi(\Delta)$ be the orbit partition induced by the lifting group on the feature index set $\mathcal{I} = \{1 \dots m\}$, let $\mathbb{R}_{[\varphi]}^m$ denote the symmetrized subspace $\{r \in \mathbb{R}^m \text{ s.t. } r_i = r_j \forall i, j \text{ in the same cell of } \varphi\}$ and define the lifted marginal polytope $\mathcal{M}_{[\varphi]}$ as $\mathcal{M} \cap \mathbb{R}_{[\varphi]}^m$, then (see Theorem 4 of [3])

$$\sup_{\mu \in \mathcal{M}} \langle \theta, \mu \rangle - A^*(\mu) = \sup_{\mu \in \mathcal{M}_{[\varphi]}} \langle \theta, \mu \rangle - A^*(\mu) \quad (3)$$

In practice, we need to work with convex variational approximations of the LHS of (3) where \mathcal{M} is relaxed to an outer bound $\text{OUTER}(\mathcal{G})$ and A^* is approximated by a convex function $B^*(\mu)$. We now state a similar result for lifting general convex approximations.

Theorem 1. *If $B^*(\mu)$ is convex and stabilized by the lifting group \mathbb{A}_Δ , i.e., for all $(\pi, \gamma) \in \mathbb{A}_\Delta$, $B^*(\mu^\gamma) = B^*(\mu)$, then φ is the lifting partition for the approximate variational problem*

$$\sup_{\mu \in \text{OUTER}(\mathcal{G})} \langle \theta, \mu \rangle - B^*(\mu) = \sup_{\mu \in \text{OUTER}_{[\varphi]}(\mathcal{G})} \langle \theta, \mu \rangle - B^*(\mu) \quad (4)$$

The importance of Theorem 1 is that it shows that it is equivalent to optimize over a subset of $\text{OUTER}(\mathcal{G})$ where pseudo-marginals in the same orbit are restricted to take the same value. As we will show in Section 4.2, this will allow us to combine many of the terms in the objective, which is where the computational gains will derive from. A

²Formally, \mathbb{G} stabilizes x if $x^g = x$ for all $g \in \mathbb{G}$.

³If $\Delta = \{\Delta_1 \dots \Delta_K\}$ is a partition of S , then each subset $\Delta_k \subset S$ is called a cell.

sketch of its proof is as follows. Consider a single pseudo-marginal vector μ . Since the objective value is the same for every μ^γ for $(\pi, \gamma) \in \mathbb{A}_\Delta$ and since the objective is concave, the *average* of these, $\frac{1}{|\mathbb{A}_\Delta|} \sum_{(\pi, \gamma) \in \mathbb{A}_\Delta} \mu^\gamma$, must have at least as good of an objective value. Furthermore, note that this averaged vector lives in the symmetrized subspace. Thus, it suffices to optimize over $\text{OUTER}_{[\varphi]}$.

4 Lifted Tree-Reweighted Problem

4.1 Symmetry of TRW Bounds

We now show that Theorem 1 can be used to lift the TRW optimization problem (1). Note that the applicability of Theorem 1 is not immediately obvious since B^* depends on the distribution over trees implicit in ρ . In establishing that the condition in Theorem 1 holds, we need to be careful so that the choice of the distribution over trees ρ does not destroy the symmetry of the problem.

The result below ensures that with no loss in optimality, ρ can be assumed to be suitably symmetric. More specifically, let $\varphi^E = \varphi^E(\Delta)$ be the set of \mathcal{G} 's edge orbits induced by the action of the lifting group \mathbb{A}_Δ ; the edge-weights ρ_e for every e in the same edge orbits can be constrained to be the same, i.e. ρ can be restricted to $\mathbb{T}_{[\varphi^E]}$.

Theorem 2. *For any $\rho \in \mathbb{T}$, there exists a symmetrized $\hat{\rho} \in \mathbb{T}_{[\varphi^E]}$ that yields at least as good an upper bound, i.e.*

$$B(\theta, \hat{\rho}) \leq B(\theta, \rho) \quad \forall \theta \in \Theta_{[\Delta]}$$

As a consequence, in optimizing the edge appearance, ρ can be restricted to the symmetrized spanning tree polytope $\mathbb{T}_{[\varphi^E]}$

$$\forall \theta \in \Theta_{[\Delta]}, \inf_{\rho \in \mathbb{T}} B(\theta, \rho) = \inf_{\rho \in \mathbb{T}_{[\varphi^E]}} B(\theta, \rho)$$

Proof. Let ρ be the argmin of the LHS, and define $\hat{\rho} = \frac{1}{|\mathbb{A}_\Delta|} \sum_{\pi \in \mathbb{A}_\Delta} \rho^\pi$ so that $\hat{\rho} \in \mathbb{T}_{[\varphi^E]}$. For all $(\pi, \gamma) \in \mathbb{A}_\Delta$ and for all tied-parameter $\theta \in \Theta_{[\Delta]}$, $\theta^\pi = \theta$, so $B(\theta, \rho^\pi) = B(\theta^\pi, \rho^\pi)$. By theorem 1 of [3], π must be an automorphism of the graph \mathcal{G} . By lemma 7 (see supplementary material), $B(\theta^\pi, \rho^\pi) = B(\theta, \rho)$. Thus $B(\theta, \rho^\pi) = B(\theta, \rho)$. Since B is convex w.r.t. ρ , by Jensen's inequality we have that $B(\theta, \hat{\rho}) \leq \frac{1}{|\mathbb{A}_\Delta|} \sum_{\pi \in \mathbb{A}_\Delta} B(\theta, \rho^\pi) = B(\theta, \rho)$. \square

Using a symmetric choice of ρ , the TRW bound B^* then satisfies the condition of theorem 1, enabling the applicability of the general lifted variational inference framework.

Theorem 3. *For a fixed $\rho \in \mathbb{T}_{[\varphi^E]}$, φ is the lifting partition for the TRW variational problem*

$$\sup_{\tau \in \text{OUTER}(\mathcal{G})} \langle \tau, \theta \rangle - B^*(\tau, \rho) = \sup_{\tau \in \text{OUTER}_{[\varphi]}(\mathcal{G})} \langle \tau, \theta \rangle - B^*(\tau, \rho) \quad (5)$$

4.2 Lifted TRW Problems

We give the explicit lifted formulation of the TRW optimization problem (5). As in [3], we restrict τ to $\text{OUTER}_{[\varphi]}$ by introducing the lifted variables $\bar{\tau}_j$ for each cell φ_j , and for all $i \in \varphi_j$, enforcing that $\tau_i = \bar{\tau}_j$. Effectively, we substitute every occurrence of $\tau_i, i \in \varphi_j$ by $\bar{\tau}_j$; in vector form, τ is substituted by $D\bar{\tau}$ where D is the characteristic matrix of the partition φ : $D_{ij} = 1$ if $i \in \varphi_j$ and 0 otherwise. This results in the lifted form of the TRW problem

$$\sup_{D\bar{\tau} \in \text{OUTER}} \langle \bar{\tau}, \bar{\theta} \rangle - \bar{B}^*(\bar{\tau}, \bar{\rho}) \quad (6)$$

where $\bar{\theta} = D^\top \theta$; \bar{B}^* is obtained from B^* via the above substitution; and $\bar{\rho}$ is the edge appearance per edge-orbit: for every edge orbit \mathbf{e} , and for every edge $e \in \mathbf{e}$, $\rho_e = \bar{\rho}_{\mathbf{e}}$. Using an alternative but equivalent form $B^* = -\sum_{v \in V} (1 - \sum_{e \in N(v)} \rho_e) H(\tau_v) - \sum_{e \in E} \rho_e H(\tau_e)$, we obtain the following explicit form for

$$\begin{aligned} \bar{B}^*(\bar{\tau}, \bar{\rho}) = & - \sum_{\mathbf{v} \in \bar{V}} \left(|\mathbf{v}| - \sum_{\mathbf{e} \in N(\mathbf{v})} |\mathbf{e}| d(\mathbf{v}, \mathbf{e}) \bar{\rho}_{\mathbf{e}} \right) H(\bar{\tau}_{\mathbf{v}}) \\ & - \sum_{\mathbf{e} \in \bar{E}} |\mathbf{e}| \bar{\rho}_{\mathbf{e}} H(\bar{\tau}_{\mathbf{e}}) \end{aligned} \quad (7)$$

Intuitively, the above can be viewed as a combination of node and edge entropies defined on nodes and edges of the lifted graph $\bar{\mathcal{G}}$. Nodes of $\bar{\mathcal{G}}$ are the node orbits of \mathcal{G} while edges are the edge-orbits of \mathcal{G} . $\bar{\mathcal{G}}$ is not a simple graph: it can have self-loops or multi-edges between the same node pair (see Fig. 1). We encode the incidence on this graph as follows: $d(\mathbf{v}, \mathbf{e}) = 0$ if \mathbf{v} is not incident to \mathbf{e} , $d(\mathbf{v}, \mathbf{e}) = 1$ if \mathbf{v} is incident to \mathbf{e} and \mathbf{e} is not a loop, $d(\mathbf{v}, \mathbf{e}) = 2$ if \mathbf{e} is a loop incident to \mathbf{v} . The *entropy* at the node orbit \mathbf{v} is defined as

$$H(\bar{\tau}_{\mathbf{v}}) = - \sum_{t \in \mathcal{X}} \bar{\tau}_{\mathbf{v}:t} \ln(\bar{\tau}_{\mathbf{v}:t})$$

and the entropy at the edge orbit \mathbf{e} is

$$H(\bar{\tau}_{\mathbf{e}}) = - \sum_{t, h \in \mathcal{X}} \bar{\tau}_{\{e_1:t, e_2:h\}} \ln(\bar{\tau}_{\{e_1:t, e_2:h\}})$$

where $\{e_1, e_2\}$ for $e_1, e_2 \in V$ is a representative (any element) of \mathbf{e} , $\{e_1:t, e_2:h\}$ is an assignment of the ground edge $\{e_1, e_2\}$, and $\{e_1:t, e_2:h\}$ is the assignment orbit. As in [3], we write $\{e_1:t, e_2:t\}$ as $\mathbf{e}:t$, and for $t < h$, $\{e_1:t, e_2:h\}$ as $\mathbf{a}:(t, h)$ where \mathbf{a} is the arc-orbit (e_1, e_2) .

When OUTER is the local or cycle polytope, the constraints $D\bar{\tau} \in \text{OUTER}$ yield the lifted local (or cycle) polytope respectively. For these constraints, we use the same form given in [3]. In section 6, we describe a set of additional constraints for further tightening when some cluster of nodes are exchangeable.

Example. Consider the MRF shown in Fig. 1 (left) with 10 binary variables that we denote B_i (for the blue nodes) and

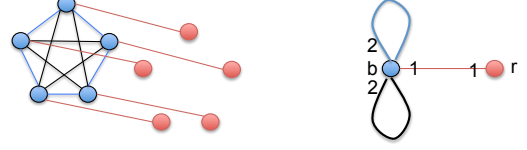


Figure 1: Left: ground graphical model. Same colored nodes and edges have the same parameters. Right: lifted graph showing 2 node orbits (\mathbf{b} and \mathbf{r}), and 3 edge orbits. Numbers on the lifted graph representing the incidence degree $d(\mathbf{v}, \mathbf{e})$ between an edge and a node orbit.

R_i (for the red nodes). The node and edge coloring denotes shared parameters. Let θ_b and θ_r be the single-node potentials used for the blue and red nodes, respectively. Let θ_{r_e} be the edge potential used for the red edges connecting the blue and red nodes, θ_{b_e} for the edge potential used for the blue edges (B_i, B_{i+1}), and θ_{k_e} for the edge potential used for the black edges (B_i, B_{i+2}).

There are two node orbits: $\mathbf{b} = \{B_1, \dots, B_5\}$ and $\mathbf{r} = \{R_1, \dots, R_5\}$. There are three edge orbits: \mathbf{r}_e for the red edges, \mathbf{b}_e for the blue edges, and \mathbf{k}_e for the black edges. The size of the node and edge orbits are all 5 (e.g., $|\mathbf{b}| = |\mathbf{b}_e| = 5$), and $d(\mathbf{b}, \mathbf{r}_e) = d(\mathbf{r}, \mathbf{r}_e) = 1$, $d(\mathbf{b}, \mathbf{b}_e) = d(\mathbf{b}, \mathbf{k}_e) = 2$. Suppose that ρ corresponds to a uniform distribution over spanning trees, which satisfies the symmetry needed by Theorem 2. We then have $\bar{\rho}_{\mathbf{r}_e} = 1$ and $\bar{\rho}_{\mathbf{b}_e} = \bar{\rho}_{\mathbf{k}_e} = \frac{2}{5}$. Putting all of this together, the lifted TRW entropy is given by $\bar{B}^*(\bar{\tau}, \bar{\rho}) = 8H(\bar{\tau}_{\mathbf{b}}) - 5H(\bar{\tau}_{\mathbf{r}_e}) - 2H(\bar{\tau}_{\mathbf{b}_e}) - 2H(\bar{\tau}_{\mathbf{k}_e})$. We illustrate the expansion of the entropy of the red edge orbit $H(\bar{\tau}_{\mathbf{r}_e})$. This edge orbit has 2 corresponding arc-orbits: $\mathbf{rb}_a = \{(R_i, B_i)\}$ and $\mathbf{br}_a = \{(B_i, R_i)\}$. Thus, $H(\bar{\tau}_{\mathbf{r}_e}) = -\bar{\tau}_{\mathbf{r}_e:00} \ln \bar{\tau}_{\mathbf{r}_e:00} - \bar{\tau}_{\mathbf{r}_e:11} \ln \bar{\tau}_{\mathbf{r}_e:11} - \bar{\tau}_{\mathbf{rb}_a:01} \ln \bar{\tau}_{\mathbf{rb}_a:01} - \bar{\tau}_{\mathbf{br}_a:01} \ln \bar{\tau}_{\mathbf{br}_a:01}$.

Finally, the linear term in the objective is given by $\langle \bar{\tau}, \bar{\theta} \rangle = 5 \langle \bar{\tau}_{\mathbf{b}}, \theta_b \rangle + 5 \langle \bar{\tau}_{\mathbf{r}}, \theta_r \rangle + 5 \langle \bar{\tau}_{\mathbf{r}_e}, \theta_{r_e} \rangle + 5 \langle \bar{\tau}_{\mathbf{b}_e}, \theta_{b_e} \rangle + 5 \langle \bar{\tau}_{\mathbf{k}_e}, \theta_{k_e} \rangle$ where, as an example, $\langle \bar{\tau}_{\mathbf{r}_e}, \theta_{r_e} \rangle = \bar{\tau}_{\mathbf{r}_e:00} \theta_{r_e,00} + \bar{\tau}_{\mathbf{r}_e:11} \theta_{r_e,11} + \bar{\tau}_{\mathbf{rb}_a:01} \theta_{r_e,01} + \bar{\tau}_{\mathbf{br}_a:01} \theta_{r_e,10}$

4.3 Optimization using Frank-Wolfe

What remains is to describe how to optimize Eq. 6. Our lifted tree-reweighted algorithm is based on Frank-Wolfe, also known as the conditional gradient method [7, 11]. First, we initialize with a pseudo-marginal vector corresponding to the uniform distribution, which is guaranteed to be in the lifted marginal polytope. Next, we solve the linear program whose objective is given by the gradient of the objective Eq. 6 evaluated at the current point, and whose constraint set is OUTER . When using the lifted cycle relaxation, we solve this linear program using a cutting-plane algorithm [3, 23]. We then perform a line search to find the optimal step size using a golden section search (a type of binary search that finds the maxima of a unimodal function), and finally repeat using the new pseudo-marginal vector. We warm start each linear program using the optimal basis found in the previous run, which makes the LP solves ex-

tremely fast after the first couple of iterations. Although we use a generic LP solver in our experiments, it is also possible to use dual decomposition to derive efficient algorithms specialized to graphical models [24].

5 Lifted Maximum Spanning Tree

Optimizing the TRW edge appearance probability ρ requires finding the maximum spanning tree (MST) in the ground graphical model. For lifted TRW, we need to perform MST while using only information from the node and edge orbits, without referring to the ground graph. In this section, we present a lifted MST algorithm for symmetric graphs which works at the orbit level.

Suppose that we are given a *weighted* graph (\mathcal{G}, w) , its automorphism group $\mathbb{A} = \text{Aut}(\mathcal{G})$ and its node and edge orbits. We aim to derive an algorithm analogous to the Kruskal’s algorithm, but with complexity depends only on the number of node/edge orbits of \mathcal{G} . However, if the algorithm has to return an actual spanning tree of \mathcal{G} then clearly its complexity cannot be less than $O(|V|)$. Instead, we consider an equivalent problem: solving a linear program on the spanning-tree polytope

$$\sup_{\rho \in \mathbb{T}(\mathcal{G})} \langle \rho, w \rangle \quad (8)$$

The same mechanism for lifting convex optimization problem (Lemma 1 in [3]) applies to this problem. Let φ^E be the edge orbit partition, then an equivalent lifted problem is

$$\sup_{\rho \in \mathbb{T}_{[\varphi^E]}} \langle \rho, w \rangle \quad (9)$$

Since ρ_e is constrained to be the same for edges in the same orbit, it is now possible to solve (9) with complexity depending only on the number of orbits. Any solution ρ of the LP (8) can be turned into a solution $\bar{\rho}$ of (9) by letting $\bar{\rho}(\mathbf{e}) = \frac{1}{|\mathbf{e}|} \sum_{e' \in \mathbf{e}} \rho(e')$.

5.1 Lifted Kruskal’s Algorithm

The Kruskal’s algorithm first sorts the edges according to their decreasing weight. Then starting from an empty graph, at each step it greedily attempts to add the next edge while maintaining the property that the used edges form a forest (containing no cycle). The forest obtained at the end of this algorithm is a maximum-weight spanning tree.

Imagine how Kruskal’s algorithm would operate on a weighted graph \mathcal{G} with non-trivial automorphisms. Let $\mathbf{e}_1, \dots, \mathbf{e}_k$ be the list of edge-orbits sorted in the order of decreasing weight (the weights w on all edges in the same orbit by definition must be the same). The main question therefore is how many edges in each edge-orbit \mathbf{e}_i will be added to the spanning tree by the Kruskal’s algorithm. Let \mathcal{G}_i be the subgraph of \mathcal{G} formed by the set of all the edges and nodes in $\mathbf{e}_1, \dots, \mathbf{e}_i$. Let $V(\mathcal{G})$ and $C(\mathcal{G})$ denote the set of nodes and set of connected components of a graph, respectively. Then (see the supplementary material for proof)

Lemma 4. *The number of edges in \mathbf{e}_i appearing in the MST found by the Kruskal’s algorithm is $\delta_V^{(i)} - \delta_C^{(i)}$ where $\delta_V^{(i)} = |V(\mathcal{G}_i)| - |V(\mathcal{G}_{i-1})|$ and $\delta_C^{(i)} = |C(\mathcal{G}_i)| - |C(\mathcal{G}_{i-1})|$. Thus a solution for the linear program (9) is $\bar{\rho}(\mathbf{e}_i) = \frac{\delta_V^{(i)} - \delta_C^{(i)}}{|\mathbf{e}_i|}$.*

5.2 Lifted Counting of the Number of Connected Components

We note that counting the number of nodes can be done simply by adding the size of each node orbit. The remaining difficulty is how to count the number of connected components of a given graph⁴ \mathcal{G} using only information at the orbit level. Let $\bar{\mathcal{G}}$ be the lifted graph of \mathcal{G} . Then (see supplementary material for proof)

Lemma 5. *If $\bar{\mathcal{G}}$ is connected then all connected components of \mathcal{G} are isomorphic. Thus if furthermore \mathcal{G}' is a connected component of \mathcal{G} then $|C(\mathcal{G})| = |V(\mathcal{G})|/|V(\mathcal{G}')|$.*

To find just one connected component, we can choose an arbitrary node u and compute $\bar{\mathcal{G}}[u]$, the lifted graph fixing u (see section 8.1 in [3]), then search for the connected component in $\bar{\mathcal{G}}[u]$ that contains $\{u\}$. Finally, if $\bar{\mathcal{G}}$ is not connected, we simply apply lemma 5 for each connected component of $\bar{\mathcal{G}}$.

The final lifted Kruskal’s algorithm combines lemma 4 and 5 while keeping track of the set of connected components of $\bar{\mathcal{G}}_i$ incrementally. The full algorithm is given in the supplementary material.

6 Tightening via Exchangeable Polytope Constraints

One type of symmetry often found in first-order probabilistic models are large sets of exchangeable random variables. In certain cases, exact inference with exchangeable variables is possible via lifted counting elimination and its generalization [17, 2]. The drawback of these exact methods is that they do not apply to many models (e.g., those with transitive clauses). Lifted variational inference methods do not have this drawback, however local and cycle relaxation can be shown to be loose in the exchangeable setting, a potentially serious limitation compared to earlier work.

To remedy this situation, we now show how to take advantage of highly symmetric subset of variables to tighten the relaxation of the lifted marginal polytope.

We call a set of random variables χ an *exchangeable* cluster iff χ can be arbitrary permuted while preserving the probability model. Mathematically, the lifting group \mathbb{A}_Δ acts on χ and the image of the action is isomorphic to $\mathbb{S}(\chi)$,

⁴Since we are only interested in connectivity in this subsection, the weights of \mathcal{G} play no role. Thus, orbits in this subsection can also be generated by the automorphism group of the unweighted version of \mathcal{G} .

the symmetric group on χ . The distribution of the random variables in χ is also exchangeable in the usual sense.

Our method for tightening the relaxation of the marginal polytope is based on lift-and-project, wherein we introduce auxiliary variables specifying the joint distribution of a large cluster of variables, and then enforce consistency between the cluster distribution and the pseudo-marginal vector [20, 24, 27]. In the ground model, one typically works with small clusters (e.g., triplets) because the number of variables grows exponentially with cluster size. The key (and nice) difference in the lifted case is that we can make use of very large clusters of highly symmetric variables: while the grounded relaxation would clearly blow up, the corresponding lifted relaxation can still remain compact.

Specifically, for an exchangeable cluster χ of arbitrary size, one can add cluster consistency constraints for the entire cluster and still maintain tractability. To keep the exposition simple, we assume that the variables are binary. Let \mathcal{C} denote a χ -configuration, i.e., a function $\mathcal{C} : \chi \rightarrow \{0, 1\}$. The set $\{\tau_{\mathcal{C}}^{\chi} \mid \forall \text{ configuration } \mathcal{C}\}$ is the collection of χ -cluster auxiliary variables. Since χ is exchangeable, all nodes in χ belong to the same node orbit; we call this node orbit $\mathbf{v}(\chi)$. Similarly, $\mathbf{e}(\chi)$ and $\mathbf{a}(\chi)$ denote the single edge and arc orbit that contains all edges and arcs in χ respectively. Let u_1, u_2 be two distinct nodes in χ . To enforce consistency between the cluster χ and the edge $\{u_1, u_2\}$ in the ground model, we introduce the constraints

$$\exists \tau^{\chi} : \sum_{\mathcal{C} \text{ s.t. } \mathcal{C}(u_i)=s_i} \tau_{\mathcal{C}}^{\chi} = \tau_{u_1:s_1, u_2:s_2} \quad \forall s_i \in \{0, 1\} \quad (10)$$

These constraints correspond to using intersection sets of size two, which can be shown to be the exact characterization of the marginal polytope involving variables in χ if the graphical model only has pairwise potentials. If higher-order potentials are present, a tighter relaxation could be obtained by using larger intersection sets together with the techniques described below.

The constraints in (10) can be methodically lifted by replacing occurrences of ground variables with lifted variables at the orbit level. First observe that in place of the grounded variables $\tau_{u_1:s_1, u_2:s_2}$, the lifted local relaxation has three corresponding lifted variables, $\bar{\tau}_{\mathbf{e}(\chi):00}$, $\bar{\tau}_{\mathbf{e}(\chi):11}$ and $\bar{\tau}_{\mathbf{a}(\chi):01}$. Second, we consider the orbits of the set of configurations \mathcal{C} . Since χ is exchangeable, there can be only $|\chi| + 1$ χ -configuration orbits; each orbit contains all configurations with precisely k 1's where $k = 0 \dots |\chi|$. Thus, instead of the $2^{|\chi|}$ ground auxiliary variables, we only need $|\chi| + 1$ lifted cluster variables. Further manipulation leads to the following set of constraints, which we call *lifted exchangeable polytope* constraints.

Theorem 6. *Let χ be an exchangeable cluster of size n ; $\mathbf{e}(\chi)$ and $\mathbf{a}(\chi)$ be the single edge and arc orbit of the graphical model that contains all edges and arcs in χ respectively; $\bar{\tau}$ be the lifted marginals. Then there exist*

$c_k^{\chi} \geq 0$, $k = 0 \dots n$ such that

$$\begin{aligned} \sum_{k=0}^{n-2} \frac{(n-k)(n-k-1)}{n(n-1)} c_k^{\chi} &= \bar{\tau}_{\mathbf{e}(\chi):00} \\ \sum_{k=0}^{n-2} \frac{(k+1)(k+2)}{n(n-1)} c_{k+2}^{\chi} &= \bar{\tau}_{\mathbf{e}(\chi):11} \\ \sum_{k=0}^{n-2} \frac{(n-k-1)(k+1)}{n(n-1)} c_{k+1}^{\chi} &= \bar{\tau}_{\mathbf{a}(\chi):01} \end{aligned}$$

Proof. See the supplementary material. \square

In contrast to the lifted local and cycle relaxations, the number of variables and constraints in the lifted exchangeable relaxation depends linearly on the domain size of the first-order model. From the lifted local constraints given by [3], $\bar{\tau}_{\mathbf{e}(\chi):00} + \bar{\tau}_{\mathbf{e}(\chi):11} + 2\bar{\tau}_{\mathbf{a}(\chi):01} = 1$. Substituting in the expression involved \bar{c}_k^{χ} , we get $\sum_{k=0}^n c_k^{\chi} = 1$. Intuitively, c_k^{χ} represents the approximation of the marginal probability $\Pr(\sum_{i \in \chi} x_i = k)$ of having precisely k ones in χ .

As proved by [2], groundings of unary predicates in Markov Logic Networks (MLNs) gives rise to exchangeable clusters. Thus, for MLNs, the above theorem immediately suggests a tightening of the relaxation: for every unary predicate of a MLN, add a new set of constraints as above to the existing lifted local (or cycle) optimization problem. Although it is not the focus of our paper, we note that this should also improve the lifted MAP inference results of [3]. For example, in the case of a symmetric complete graphical model, lifted MAP inference using the linear program given by these new constraints would find the exact k that maximizes $\Pr(x_{\chi})$, hence recover the same solution as counting elimination. Marginal inference may still be inexact due to the tree-reweighted entropy approximation. We re-emphasize that the complexity of variational inference with lifted exchangeable constraints is guaranteed to be polynomial in the domain size, unlike exact methods based on lifted counting elimination and variable elimination.

7 Experiments

In this section, we provide an empirical evaluation of our lifted tree reweighted (LTRW) algorithm. As a baseline we use a dampened version of the lifted belief propagation (LBP-Dampening) algorithm from [21]. Our lifted algorithm has all of the same advantages of the tree-reweighted approach over belief propagation, which we will illustrate in the results: (1) a convex objective that can be convergently solved to optimality, (2) upper bounds on the partition function, and (3) the ability to easily improve the approximation by tightening the relaxation. Our evaluation includes four variants of the LTRW algorithm corresponding to using different outer bounds: lifted local polytope (LTRW-L), lifted cycle polytope (LTRW-C), lifted local

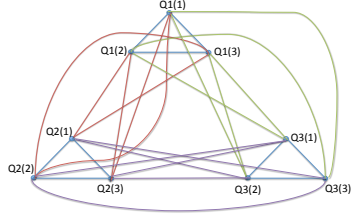


Figure 2: An example of the ground graphical model for the Clique-Cycle model (domain size = 3).

polytope with exchangeable polytope constraints (LTRW-LE), and lifted cycle polytope with exchangeable constraints (LTRW-CE). The conditional gradient optimization of the lifted TRW objective terminates when the duality gap is less than 10^{-4} or when a maximum number of 1000 iterations is reached. To solve the LP problem during conditional gradient, we use Gurobi⁵.

We evaluate all the algorithms using several first-order probabilistic models. We assume that no evidence has been observed, which results in a large amount of symmetry. Even without evidence, performing marginal inference in first-order probabilistic models can be very useful for maximum likelihood learning [13]. Furthermore, the fact that our lifted tree-reweighted variational approximation provides an upper bound on the partition function enables us to maximize a lower bound on the likelihood [29], which we demonstrate in Sec. 7.5. To find the lifted orbit partition, we use the renaming group as in [3] which exploits the symmetry of the unobserved contents in the model.

Rather than optimize over the spanning tree polytope, which is computationally intensive, most TRW implementations use a single fixed choice of edge appearance probabilities, e.g. an (un)weighted distribution obtained using the matrix-tree theorem. In these experiments, we initialize the lifted edge appearance probabilities $\bar{\rho}$ to be the most uniform per-orbit edge-appearance probabilities by solving the optimization problem $\inf_{\bar{\rho} \in \mathbb{T}_{[\varphi, E]}} (\bar{\rho} - \frac{|V|-1}{|E|})^2$ using conditional gradient. Each direction-finding step of this conditional gradient solves a lifted MST problem of the form $\sup_{\bar{\rho}' \in \mathbb{T}_{[\varphi, E]}} \left\langle -2(\bar{\rho} - \frac{|V|-1}{|E|}), \bar{\rho}' \right\rangle$ using our lifted Kruskal’s algorithm, where $\bar{\rho}$ is the current solution. After this initialization, we fix the lifted edge appearance probabilities and do not attempt to optimize them further.

7.1 Test models

Fig. 3 describes the four test models in MLN syntax. We focus on the repulsive case, since for attractive models, all TRW variants and lifted LBP give similar results. The parameter W denotes the weight that will be varying during the experiments. In all models except *Clique-Cycle*, W acts like the “local field” potential in an Ising model; a negative (or positive) value of W means the corresponding variable tends to be in the 0 (or 1) state. *Complete-*

Graph is equivalent to an Ising model on the complete graph of size n (the domain size) with homogenous parameters. Exact marginals and the log-partition function can be computed in closed form using lifted counting elimination. The weight of the interaction clause is set to -0.1 (repulsive). *Friends-Smokers (negated)* is a variant of the Friends-Smokers model [21] where the weight of the final clause is set to -1.1 (repulsive). We use the method in [2] to compute the exact marginal for the *Cancer* predicate and the exact value of the log-partition function. *Lovers-Smokers* is the same MLN used in [3] with a full transitive clause and where we vary the prior of the *Loves* predicate. *Clique-Cycle* is a model with 3 cliques and 3 bipartite graphs in between. Its corresponding ground graphical model is shown in Fig. 2.

7.2 Accuracy of Marginals

Fig. 4 shows the marginals computed by all the algorithms as well as exact marginals on the Complete-Graph and Friends-Smokers models. We do not know how to efficiently perform exact inference in the remaining two models, and thus do not measure accuracy for them. The result on complete graphs illustrates the clear benefit of tightening the relaxation: LTRW-Local and LBP are inaccurate for moderate W , whereas cycle constraints and, especially, exchangeable constraints drastically improve accuracy. As discussed earlier, for the case of symmetric complete graphical models, the exchangeable constraints suffice to exactly characterize the marginal polytope. As a result, the approximate marginals computed by LTRW-LE and LTRW-CE are almost the same as the exact marginals; the very small difference is due to the entropy approximation. On the Friends-Smokers (negated) model, all LTRW variants give accurate marginals while lifted LBP even with very strong dampening (0.9 weight given to previous iterations’ messages) fails to converge for $W < 2$. We observed that LTRW-LE gives the best trade-off between accuracy and running time for this model. Note that we do not compare to ground versions of the lifted TRW algorithms because, by Theorem 3, the marginals and log-partition function are the same for both.

7.3 Quality of Log-Partition Upper bounds

Fig. 5 plots the values of the upper bounds obtained by the LTRW algorithms on the four test models. The results clearly show the benefits of adding each type of constraint to the LTRW, with the best upper bound obtained by tightening the lifted local polytope with both lifted cycle and exchangeable constraints. For the Complete-Graph and Friends-Smokers model, the log-partition approximation using exchangeable polytope constraints is very close to exact. In addition, we illustrate lifted LBP’s approximation of the log-partition function on the Complete-Graph (note it is non-convex and not an upper bound).

⁵<http://www.gurobi.com/>

Complete Graph		Friends-Smokers (Negated)	
W	$V(x)$	W	$[x \neq y \wedge \neg \text{Friends}(x, y)]$
-0.1	$[x \neq y \wedge (V(x) \Leftrightarrow V(y))]$	1.4	$\neg \text{Smokes}(x)$
		2.3	$\neg \text{Cancer}(x)$
		1.5	$\text{Smokes}(x) \Rightarrow \text{Cancer}(x)$
		-1.1	$[x \neq y \wedge \text{Smokes}(x) \wedge \text{Friends}(x, y) \Rightarrow \text{Smokes}(y)]$
Lovers-Smokers		Clique-Cycle	
W	$[x \neq y \wedge \text{Loves}(x, y)]$	W	$x \neq y \wedge (Q1(x) \Leftrightarrow \neg Q2(y))$
100	$\text{Male}(x) \Leftrightarrow \neg \text{Female}(x)$	W	$x \neq y \wedge (Q2(x) \Leftrightarrow \neg Q3(y))$
2	$\text{Male}(x) \wedge \text{Smokes}(x)$	W	$x \neq y \wedge (Q3(x) \Leftrightarrow \neg Q1(y))$
1	$\text{Female}(x) \wedge \text{Smokes}(x)$	$-W$	$x \neq y \wedge (Q1(x) \Leftrightarrow Q1(y))$
0.5	$[x \neq y \wedge \text{Male}(x) \wedge \text{Female}(y) \wedge \text{Loves}(x, y)]$	$-W$	$x \neq y \wedge (Q2(x) \Leftrightarrow Q2(y))$
1	$[x \neq y \wedge \text{Loves}(x, y) \wedge (\text{Smokes}(x) \Leftrightarrow \text{Smokes}(y))]$	$-W$	$x \neq y \wedge (Q3(x) \Leftrightarrow Q3(y))$
-100	$[x \neq y \wedge y \neq z \wedge z \neq x \wedge \text{Loves}(x, y) \wedge \text{Loves}(y, z) \wedge \text{Loves}(x, z)]$		

Figure 3: Test models

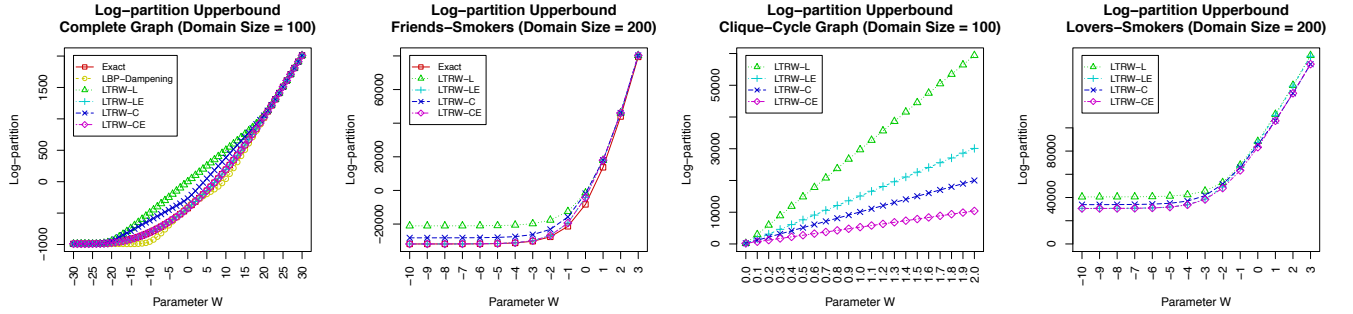


Figure 5: Approximations of the log-partition function on the four test models from Fig. 3 (best viewed in color).

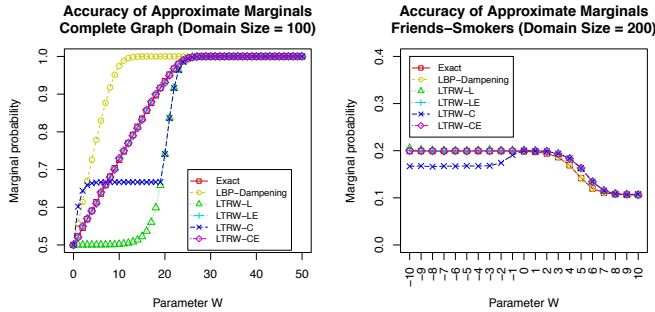


Figure 4: Left: marginal accuracy for complete graph model. Right: marginal accuracy for $Pr(\text{Cancer}(x))$ in Friends-Smokers (neg). Lifted TRW variants using different outer bounds: L=local, C=cycle, LE=local+exchangeable, CE=cycle+exchangeable (best viewed in color).

Domain size	10	20	30	100	200
TRW-L	138370	609502	1525140	-	-
LTRW-L	3255	3581	3438	1626	1416
LTRW-LE	681	703	721	1033	1307

Table 1: Ground vs lifted TRW runtime on Complete-Graph (milliseconds)

meaningful timing comparison with LBP. For example, LBP did not converge for about half of the values of W in the *Lovers-Smokers* model, even after using very strong dampening. We did observe that when LBP converges, it is much faster than LTRW. We hypothesize that this is due to the message passing nature of LBP, which is based on a fixed point update whereas our algorithm is based on Frank-Wolfe.

7.4 Running time

As shown in Table 1, lifted variants of TRW are order-of-magnitudes faster than the ground version. Interestingly, lifted TRW with local constraints is observed to be faster as the domain size increases; this is probably due to the fact that as the domain size increases, the distribution becomes more peak, so marginal inference becomes more similar to MAP inference. Lifted TRW with local and exchangeable constraints requires a smaller number of conditional gradient iterations, thus is faster; however note that its running time slowly increases since the exchangeable constraint set grows linearly with domain size.

LBP's lack of convergence makes it difficult to have a

7.5 Application to Learning

We now describe an application of our algorithm to the task of learning relational Markov networks for inferring protein-protein interactions from noisy, high-throughput, experimental assays [12]. This is equivalent to learning the parameters of an exponential family random graph model [19] where edges in the random graph represent the protein-protein interactions. Despite fully observed data, maximum likelihood learning is challenging because of the intractability of computing the log-partition function and its gradient. In particular, this relational Markov network has over 330K random variables (one for each possible interaction of 813 variables) and tertiary potentials. However, Jaimovich et al. [13] observed that the partition function in

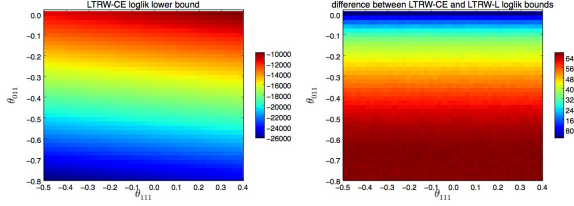


Figure 6: Log-likelihood lower-bound obtained using lifted TRW with the cycle and exchangeable constraints (CE) for the same protein-protein interaction data used in [13] (left) (c.f. Fig. 7 in [13]). Improvement in lower-bound after tightening the local constraints (L) with CE (right).

relational Markov networks is highly symmetric, and use lifted LBP to efficiently perform approximate learning in running time that is independent of the domain size. They use their lifted inference algorithm to visualize the (approximate) likelihood landscape for different values of the parameters, which among other uses characterizes the robustness of the model to parameter changes.

We use precisely the same procedure as [13], substituting lifted BP with our new lifted TRW algorithms. The model has three parameters: θ_1 , used in the single-node potential to specify the prior probability of a protein-protein interaction; θ_{111} , part of the tertiary potentials which encourages cliques of three interacting proteins; and θ_{011} , also part of the tertiary potentials which encourages chain-like structures where proteins A, B interact, B, C interact, but A and C do not (see supplementary material for the full model specification as an MLN). We follow their two-step estimation procedure, first estimating θ_1 in the absence of the other parameters (the maximum likelihood, BP, and TRW estimates of this parameter coincide, and estimation can be performed in closed-form: $\theta_1^* = -5.293$). Next, for each setting of θ_{111} and θ_{011} we estimate the log-partition function using lifted TRW with the cycle+exchangeable vs. local constraints only. Since TRW is an upper bound on the log-partition function, these provide lower bounds on the likelihood.

Our results are shown in Fig. 6, and should be compared to Fig. 7 of [13]. The overall shape of the likelihood landscapes are similar. However, the lifted LBP estimates of the likelihood have several local optima, which cause gradient-based learning with lifted LBP to reach different solutions depending on the initial setting of the parameters. In contrast, since TRW is convex, any gradient-based procedure would reach the global optima, and thus learning is much easier. Interestingly, we see that our estimates of the likelihood have a significantly smaller range over these parameter settings than that estimated by lifted LBP. Moreover, the high-likelihood parameter settings extends to larger values of θ_{111} . For all algorithms there is a sudden decrease in the likelihood at $\theta_{011} > 0$ (not shown in the figure).

8 Discussion and Conclusion

Lifting partitions used by lifted and counting BP [21, 16] can be coarser than orbit partitions. In graph-theoretic

terms, these partitions are called *equitable* partitions. If each equitable partition cell is thought of as a distinct node color, then among nodes with the same color, their neighbors must have the same color histogram. It is known that orbit partitions are always equitable, however the converse is not always true [9].

Since equitable partition can be computed more efficiently and potentially leads to more compact lifted problems, the following question naturally arises: can we use equitable partition in lifting the TRW problem? Unfortunately, a complete answer is non-trivial. We point out here a theoretical barrier due to the interplay between the spanning tree polytope and the equitable partition of a graph.

Let ε be the coarsest equitable partition of edges of \mathcal{G} . We give an example graph in the supplementary material (see example 9) where the symmetrized spanning tree polytope corresponding to the equitable partition ε , $\mathbb{T}_{[\varepsilon]} = \mathbb{T}(\mathcal{G}) \cap \mathbb{R}_{[\varepsilon]}^{|E|}$ is an empty set. When $\mathbb{T}_{[\varepsilon]}$ is empty, the consequence is that if we want ρ to be within \mathbb{T} so that $B(\cdot, \rho)$ is guaranteed to be a convex upper bound of the log-partition function, we cannot restrict ρ to be consistent with the equitable partition. In lifted and counting BP, $\rho \equiv 1$ so it is clearly consistent with the equitable partition; however, one loses convexity and upper bound guarantee as a result. This suggests that there might be a trade-off between the compactness of the lifting partition and the quality of the entropy approximation, a topic deserving the attention of future work.

In summary, we presented a formalization of lifted marginal inference as a convex optimization problem and showed that it can be efficiently solved using a Frank-Wolfe algorithm. Compared to previous lifted variational inference algorithms, in particular lifted belief propagation, our approach comes with convergence guarantees, upper bounds on the partition function, and the ability to improve the approximation (e.g. by introducing additional constraints) at the cost of small additional running time.

A limitation of our lifting method is that as the amount of soft evidence (the number of distinct individual objects) approaches the domain size, the behavior of lifted inference approaches ground inference. The wide difference in running time between ground and lifted inference suggests that significant efficiency can be gained by solving an approximation of the original problem that is more symmetric [25, 15, 22, 6]. One of the most interesting open questions raised by our work is how to use the variational formulation to perform approximate lifting. Since our lifted TRW algorithm provides an upper bound on the partition function, it is possible that one could use the upper bound to guide the choice of approximation when deciding how to re-introduce symmetry into an inference task.

Acknowledgements: Work by DS supported by DARPA PPAML program under AFRL contract no. FA8750-14-C-0005.

References

- [1] F. Barahona and A. R. Mahjoub. On the cut polytope. *Mathematical Programming*, 36:157–173, 1986.
- [2] Hung Hai Bui, Tuyen N. Huynh, and Rodrigo de Salvo Braz. Lifted inference with distinct soft evidence on every object. In *AAAI-2012*, 2012.
- [3] Hung Hai Bui, Tuyen N. Huynh, and Sebastian Riedel. Automorphism groups of graphical models and lifted variational inference. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence, UAI-2013*. AUAI Press, 2013.
- [4] Hung Hai Bui, Tuyen N. Huynh, and David Sontag. Lifted tree-reweighted variational inference. *arXiv*, 2014.
- [5] R. de Salvo Braz, E. Amir, and D. Roth. Lifted first-order probabilistic inference. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI '05)*, pages 1319–1125, 2005.
- [6] Rodrigo de Salvo Braz, Sriraam Natarajan, Hung Bui, Jude Shavlik, and Stuart Russell. Anytime lifted belief propagation. In *6th International Workshop on Statistical Relational Learning (SRL 2009)*, 2009.
- [7] Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, 1956. ISSN 1931-9193.
- [8] A. Globerson and T. Jaakkola. Convergent Propagation Algorithms via Oriented Trees. In *Uncertainty in Artificial Intelligence*, 2007.
- [9] Chris Godsil and Gordon Royle. *Algebraic Graph Theory*. Springer, 2001.
- [10] Vibhav Gogate and Pedro Domingos. Probabilistic theorem proving. In *Proceedings of the Twenty-Seventh Annual Conference on Uncertainty in Artificial Intelligence (UAI-11)*, pages 256–265, 2011.
- [11] Martin Jaggi. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In *Proceedings of the 30th ICML*, volume 28, pages 427–435. JMLR Workshop and Conference Proceedings, 2013.
- [12] Ariel Jaimovich, Gal Elidan, Hanah Margalit, and Nir Friedman. Towards an integrated protein-protein interaction network: a relational markov network approach. *Journal of Computational Biology*, 13(2):145–164, 2006.
- [13] Ariel Jaimovich, Ofer Meshi, and Nir Friedman. Template based inference in symmetric relational markov random fields. In *Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence, Vancouver, BC, Canada, July 19-22, 2007*, pages 191–199. AUAI Press, 2007.
- [14] Jeremy Jancsary and Gerald Matz. Convergent decomposition solvers for tree-reweighted free energies. *Journal of Machine Learning Research - Proceedings Track*, 15:388–398, 2011.
- [15] K. Kersting, Y. El Massaoudi, B. Ahmadi, and F. Hadiji. Informed lifting for message-passing. In D. Poole M. Fox, editor, *Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10)*, Atlanta, USA, July 11 – 15 2010. AAAI Press.
- [16] Kristian Kersting, Babak Ahmadi, and Sriraam Natarajan. Counting belief propagation. In *Proceedings of the 25th Annual Conference on Uncertainty in AI (UAI '09)*, 2009.
- [17] B. Milch, L. S. Zettlemoyer, K. Kersting, M. Haimes, and L. P. Kaelbling. Lifted Probabilistic Inference with Counting Formulas. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI '08)*, pages 1062–1068, 2008.
- [18] Mathias Niepert. Markov chains on orbits of permutation groups. In *UAI-2012*, 2012.
- [19] Garry Robins, Pip Pattison, Yuval Kalish, and Dean Lusher. An introduction to exponential random graph (p^*) models for social networks. *Social networks*, 29(2):173–191, 2007.
- [20] H. D. Sherali and W. P. Adams. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM Journal on Discrete Mathematics*, 3(3):411–430, 1990. doi: 10.1137/0403036. URL <http://link.aip.org/link/?SJD/3/411/1>.
- [21] Parag Singla and Pedro Domingos. Lifted first-order belief propagation. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI '08)*, pages 1094–1099, 2008.
- [22] Parag Singla, Aniruddh Nath, and Pedro Domingos. Approximate lifted belief propagation. In *Workshop on Statistical Relational Artificial Intelligence (StaR-AI 2010)*, 2010.
- [23] D. Sontag and T. Jaakkola. New outer bounds on the marginal polytope. In *Advances in Neural Information Processing Systems 21*. MIT Press, 2008.
- [24] David Sontag, T. Meltzer, A. Globerson, T. Jaakkola, and Y. Weiss. Tightening LP relaxations for MAP using message passing. In *Proceedings of the 24th Annual Conference on Uncertainty in AI (UAI '08)*, 2008.
- [25] Guy Van den Broeck and Adnan Darwiche. On the complexity and approximation of binary evidence in lifted inference. In *Advances in Neural Information Processing Systems*, pages 2868–2876, 2013.
- [26] M. Wainwright, T. Jaakkola, and A. Willsky. A new class of upper bounds on the log partition function. *IEEE Transactions on Information Theory*, 51:2313–2335, 2005.
- [27] Martin Wainwright and Michael Jordan. *Graphical Models, Exponential Families, and Variational Inference*. Now Publishers, 2008.
- [28] Martin J. Wainwright, Tommi Jaakkola, and Alan S. Willsky. Tree-based reparameterization framework for analysis of sum-product and related algorithms. *IEEE Transactions on Information Theory*, 49:1120–1146, 2003.
- [29] C. Yanover, O. Schueler-Furman, and Y. Weiss. Minimizing and learning energy functions for side-chain prediction. *Journal of Computational Biology*, 15(7):899–911, 2008.

Approximate Decentralized Bayesian Inference

Trevor Campbell
LIDS, MIT
Cambridge, MA 02139
tdjc@mit.edu

Jonathan P. How
LIDS, MIT
Cambridge, MA 02139
jhow@mit.edu

Abstract

This paper presents an approximate method for performing Bayesian inference in models with conditional independence over a decentralized network of learning agents. The method first employs variational inference on each individual learning agent to generate a local approximate posterior, the agents transmit their local posteriors to other agents in the network, and finally each agent combines its set of received local posteriors. The key insight in this work is that, for many Bayesian models, approximate inference schemes destroy symmetry and dependencies in the model that are crucial to the correct application of Bayes' rule when combining the local posteriors. The proposed method addresses this issue by including an additional optimization step in the combination procedure that accounts for these broken dependencies. Experiments on synthetic and real data demonstrate that the decentralized method provides advantages in computational performance and predictive test likelihood over previous batch and distributed methods.

1 INTRODUCTION

Recent trends in the growth of datasets, and the methods by which they are collected, have led to increasing interest in the parallelization of machine learning algorithms. Parallelization results in reductions in both the memory usage and computation time of learning, and allows data to be collected by a network of learning agents rather than by a single central agent. There are two major classes of parallelization algorithms: those that require a globally shared memory/computation unit (e.g., a central fusion processor that each learning agent is in communication with, or the main thread on a multi-threaded computer), and those that do not. While there is as of yet no consensus in the litera-

ture on the terminology for these two types of parallelization, in this work we refer to these two classes, respectively, as *distributed* and *decentralized* learning.

Some recent approaches to distributed learning have involved using streaming variational approximations (Broderrick et al., 2013), parallel stochastic gradient descent (Niu et al., 2011), the Map-Reduce framework (Dean and Ghemawat, 2004), database-inspired concurrency control (Pan et al., 2013), and message passing on graphical models (Gonzalez et al., 2009). When a reliable central learning agent with sufficient communication bandwidth is available, such distributed learning techniques are generally preferred to decentralized learning. This is a result of the consistent global model shared by all agents, with which they can make local updates without the concern of generating conflicts unbeknownst to each other.

Decentralized learning is a harder problem in general, due to asynchronous communication/computation, a lack of a globally shared state, and potential network and learning agent failure, all of which may lead to inconsistencies in the model possessed by each agent. Addressing these issues is particularly relevant to mobile sensor networks in which the network structure varies over time, agents drop out and are added dynamically, and no single agent has the computational or communication resources to act as a central hub during learning. Past approaches to decentralized learning typically involve each agent communicating frequently to form a consensus on the model over the network, and are often model-specific: particle filtering for state estimation (Rosencrantz et al., 2003) involves sending particle sets and informative measurements to peers; distributed EM (Wolfe et al., 2008) requires communication of model statistics to the network after each local iteration; distributed Gibbs sampling (Newman et al., 2007) involves model synchronization after each sampling step; robust distributed inference (Paskin and Guestrin, 2004) requires the formation of a spanning tree of nodes in the network and message passing; asynchronous distributed learning of topic models (Asuncion et al., 2008) requires communication of model statistics to peers after each local

sampling step; and hyperparameter consensus (Fraser et al., 2012) requires using linear network consensus on exponential family hyperparameters.

The method proposed in the present paper takes a different tack; each agent computes an approximate factorized variational posterior using only their local datasets, sends and receives statistics to and from other agents in the network asynchronously, and combines the posteriors locally on-demand. Building upon insights from previous work on distributed and decentralized inference (Broderick et al., 2013, Rosencrantz et al., 2003), a naïve version of this algorithm based on Bayes’ rule is presented. It is then shown that, due to the approximation used in variational inference, this algorithm leads to poor decentralized posterior approximations for unsupervised models with inherent symmetry. Next, building on insights gained from the results of variational and Gibbs sampling inference on a synthetic example, an approximate posterior combination algorithm is presented that accounts for symmetry structure in models that the naïve algorithm is unable to capture. The proposed method is highly flexible, as it can be combined with past streaming variational approximations (Broderick et al., 2013, Lin, 2013), agents can share information with only subsets of the network, the network may be dynamic with unknown topology, and the failure of individual learning agents does not affect the operation of the rest of the network. Experiments on a mixture model, latent Dirichlet allocation (Blei et al., 2003), and latent feature assignment (Griffiths and Ghahramani, 2005) demonstrate that the decentralized method provides advantages in model performance and computational time over previous approaches.

2 APPROXIMATE DECENTRALIZED BAYESIAN INFERENCE

2.1 THE NAÏVE APPROACH

Suppose there is a set of learning agents i , $i = 1, \dots, N$, each with a distribution on a set of latent parameters θ_j , $j = 1, \dots, K$ (all parameters θ_j may generally be vectors). Suppose a fully factorized exponential family distribution has been used to approximate each agent’s posterior $q_i(\theta_1, \dots, \theta_K)$. Then the distribution possessed by each agent i is

$$q_i(\theta_1, \dots, \theta_K) = \prod_j q_{\lambda_{ij}}(\theta_j), \quad (1)$$

where λ_{ij} parameterizes agent i ’s distribution over θ_j . Given the prior

$$q_0(\theta_1, \dots, \theta_K) = \prod_j q_{\lambda_{0j}}(\theta_j), \quad (2)$$

is known by all agents, and the conditional independence of data given the model, the overall posterior distribu-

tion $q(\theta_1, \dots, \theta_K)$ may be approximated by using Bayes’ rule (Broderick et al., 2013) and summing over the λ_{ij} :

$$\begin{aligned} q(\cdot) &\propto q_0(\cdot)^{1-N} \prod_i q_i(\cdot) \\ &= \left(\prod_j q_{\lambda_{0j}}(\theta_j) \right)^{1-N} \prod_i \prod_j q_{\lambda_{ij}}(\theta_j) \\ \therefore q(\cdot) &= \prod_j q_{\lambda_j}(\theta_j) \end{aligned} \quad (3)$$

$$\text{where } \lambda_j = (1 - N)\lambda_{0j} + \sum_i \lambda_{ij}.$$

The last line follows from the use of exponential family distributions in the variational approximation. This procedure is decentralized, as each agent can asynchronously compute its individual posterior approximation, broadcast it to the network, receive approximations from other agents, and combine them locally. Furthermore, this procedure can be made to handle streaming data by using a technique such as SDA-Bayes (Broderick et al., 2013) or sequential variational approximation (Lin, 2013) on each agent locally to generate the streaming local posteriors q_i .

As an example, this method is now applied to decentralized learning of a Gaussian model with unknown mean $\mu = 1.0$ and known variance $\sigma^2 = 1.0$. The prior on μ is Gaussian with mean $\mu_0 = 0.0$ and variance $\sigma_0^2 = 2.0$. There are 10 learning agents, each of whom receives 10 observations $y \sim \mathcal{N}(\mu, \sigma^2)$. Because the Gaussian distribution is in the exponential family, the variational approximation is exact in this case. As shown in Figure 1, the decentralized posterior is the same as the batch posterior. Note that if the approximation is used on a more complicated distribution not in the exponential family, then the batch posterior may in general differ from the decentralized posterior; however, they will both approximate the same true posterior distribution.

2.2 FAILURE OF THE NAÏVE APPROACH UNDER PARAMETER PERMUTATION SYMMETRY

As a second example, we apply decentralized inference to a Gaussian mixture model with three components having unknown means $\mu = (1.0, -1.0, 3.0)$ and cluster weights $\pi = (0.6, 0.3, 0.1)$ with known variance $\sigma^2 = 0.09$. The prior on each mean μ_i was Gaussian with mean $\mu_0 = 0.0$ and variance $\sigma_0^2 = 2.0$, while the prior on the weights π was Dirichlet with parameters $(1.0, 1.0, 1.0)$. First, the true posterior, shown in Figure 2a, was formed using 30 datapoints that were sampled from the generative model. Then, the decentralized variational inference procedure in (3) was run with 10 learning agents, each of whom received 3 of the datapoints, resulting in the approximate decentralized posterior in Figure 2b.

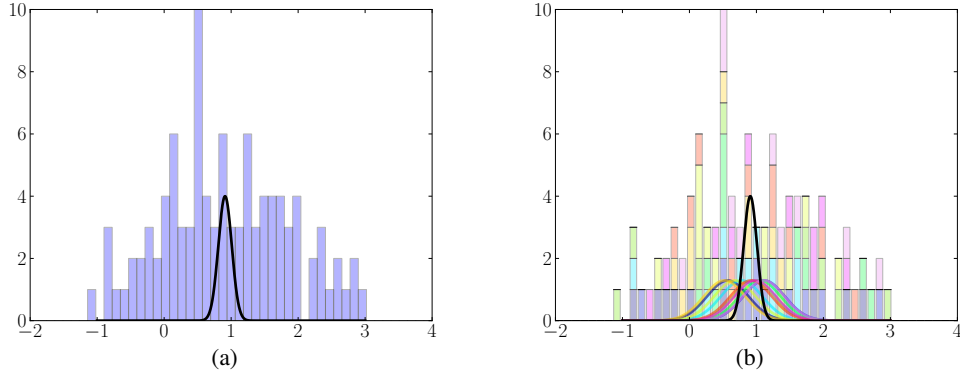


Figure 1: (1a): Batch posterior of μ in black, with histogram of observed data. (1b): Decentralized posterior of μ in black, individual posteriors in color and correspondingly colored histogram of observed data.

The decentralized posterior, in this case, is a very poor approximation of the true batch posterior. The reason for this is straightforward: approximate inference algorithms, such as variational inference with a fully factorized distribution, often do not capture *parameter permutation symmetry* in the posterior. Parameter permutation symmetry is a property of a Bayesian model in which permuting the values of some subset of the parameters does not change the posterior probability. For example, in the Gaussian mixture model, the true posterior over π, μ given data y is invariant to transformation by any permutation matrix P :

$$p(P\pi, P\mu|y) = p(\pi, \mu|y). \quad (4)$$

Indeed, examining the true posterior in Figure 2a, one can identify 6 differently colored regions; each of these regions corresponds to one of the possible $3! = 6$ permutation matrices P . In other words, the true posterior captures the invariance of the distribution to reordering of the parameters correctly.

To demonstrate that approximate inference algorithms typically do not capture parameter permutation symmetry in a model, consider the same mixture model, learned with 30 datapoints in a single batch using Gibbs sampling and variational Bayesian inference. Samples from 5 random restarts of each method are shown in Figure 3. Both algorithms fail to capture the permutation symmetry in the mixture model, and converge to one of the 6 possible orderings of the parameters. This occurs in variational Bayesian inference and Gibbs sampling for different reasons: Gibbs sampling algorithms often get stuck in local posterior likelihood optima, while the variational approximation explicitly breaks the dependence of the parameters on one another.

In a batch setting, this does not pose a problem, because practitioners typically find the selection of a particular parameter ordering acceptable. However, in the decentralized setting, this causes problems when combining the posteriors of individual learning agents. If each agent effectively picks a parameter ordering at random when per-

forming inference, combining the posteriors without considering those orderings can lead to poor results (such as that presented in Figure 2b). Past work dealing with this issue has focused primarily on modifying the samples of MCMC algorithms by introducing “identifiability constraints” that control the ordering of parameters (Jasra et al., 2005, Stephens, 2000), but these approaches are generally model-specific and restricted to use on very simple mixture models.

2.3 MERGING POSTERiors WITH PARAMETER PERMUTATION SYMMETRY

This section presents a method for locally combining the individual posteriors of decentralized learning agents when the model contains parameter permutation symmetry. Formally, suppose that the true posterior probability of $\theta_1, \dots, \theta_K$ is invariant to permutations of the components of one or more θ_j . In general, there may be subsets of parameters which have coupled symmetry, in that the true posterior is only invariant if the components of all parameters in the subset are permuted in the same way (for example, the earlier Dirichlet mixture model had coupled permutation symmetry in μ and π). It is assumed that any such coupling in the model is known beforehand by all agents. Because the exponential family variational approximation is completely decoupled, it is possible to treat each coupled permutation symmetry set of parameters in the model independently; therefore, we assume below that $\theta_1, \dots, \theta_K$ all have coupled permutation symmetry, for simplicity in the notation and exposition.

In order to properly combine the approximate posterior produced by each learning agent, first the individual posteriors are *symmetrized* (represented by a tilde) by summing over all possible permutations as follows:

$$\tilde{q}_i(\cdot) \propto \sum_P \prod_j q_{P\lambda_{ij}}(\theta_j), \quad (5)$$

where the sum is taken to be over all permutation matrices

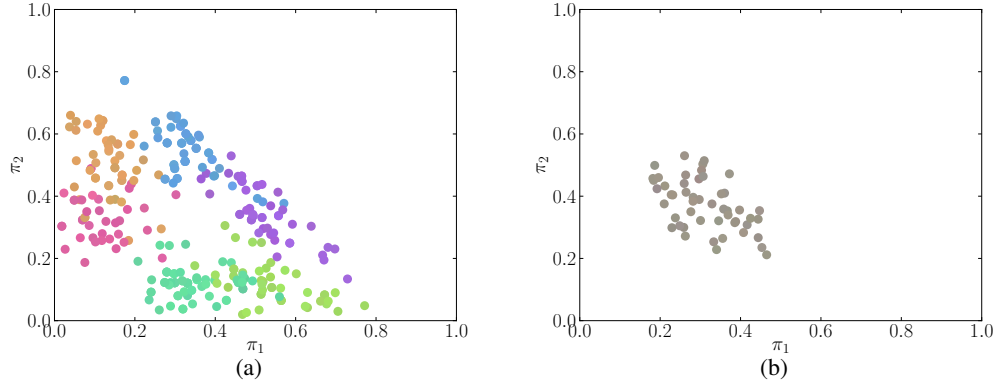


Figure 2: (2a): Samples from the true posterior over μ, π . Each particle’s position on the simplex (with $\pi_3 = 1 - \pi_1 - \pi_2$) represents the sampled weights, while RGB color coordinates of each particle represent the sampled position of the three means. (2b): Samples from the naively constructed decentralized approximate posterior, with the same coloring scheme. Note the disparity with Figure 2a.

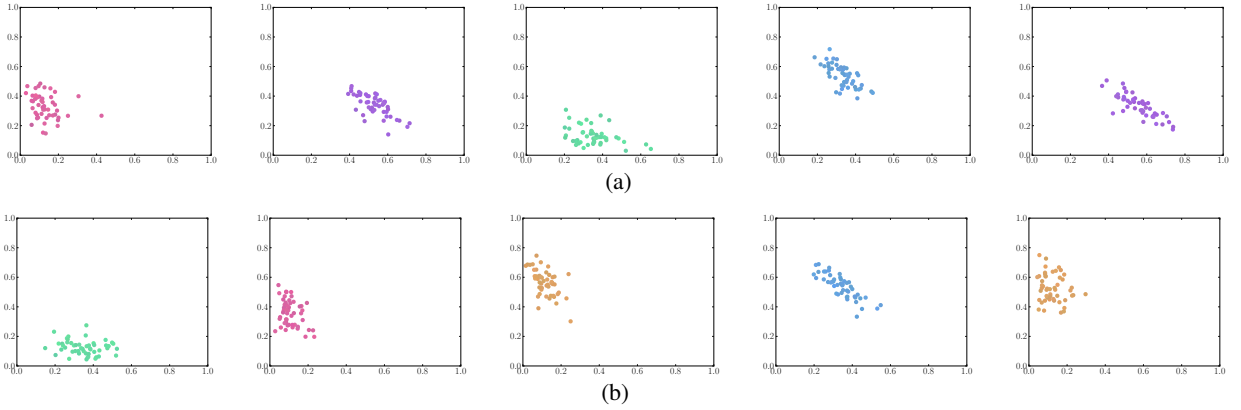


Figure 3: Batch Gibbs sampling (3a) and variational Bayes (3b) approximate posterior samples from 5 random restarts. Comparison to Figure 2a shows that both approximate inference algorithms tend to converge to a random component of the permutation symmetry in the true posterior.

P with the same dimension as λ_{ij} . This process of approximating the true single-agent posterior is referred to as symmetrization because \tilde{q}_i has the same parameter permutation symmetry as the true posterior, i.e. for all permutation matrices P ,

$$\tilde{q}_i(P\theta_1, \dots, P\theta_K) = \tilde{q}_i(\dots). \quad (6)$$

To demonstrate the effect of this procedure, the mixture model example was rerun with batch variational Bayesian inference (i.e. all 30 datapoints were given to a single learner) followed by symmetrization. Samples generated from these new symmetrized posterior distributions over 5 random restarts of the inference procedure are shown in Figure 4. This result demonstrates that the symmetrized distributions are invariant to the random permutation to which the original approximate posterior converged.

It is now possible to combine the individual (symmetrized) posteriors via the procedure outlined in (3):

$$q(\cdot) \propto q_0(\cdot)^{1-N} \prod_i \tilde{q}_i(\cdot)$$

$$\begin{aligned} &= \left(\prod_j q_{\lambda_{0j}}(\theta_j) \right)^{1-N} \prod_i \sum_{P_i} \prod_j q_{P_i \lambda_{ij}}(\theta_j) \quad (7) \\ &= \sum_{\{P_i\}_i} \prod_j \left[q_{\lambda_{0j}}(\theta_j)^{1-N} \prod_i q_{P_i \lambda_{ij}}(\theta_j) \right], \end{aligned}$$

where the outer sum is now over unique combinations of the set of permutation matrices $\{P_i\}_i$ used by the learning agents.

2.4 AMPS - APPROXIMATE MERGING OF POSTERIOR WITH SYMMETRY

The posterior distribution in (7) is unfortunately intractable to use for most purposes, as it contains a number of terms that is factorial in the dimensions of the parameters, and exponential in the number of learning agents. Therefore, we approximate this distribution by finding the component with the highest weight – the intuitive reasoning for this is that the component with the highest weight is the one for which the individual posteriors have correctly aligned permutations, thus contributing to each other the most and rep-

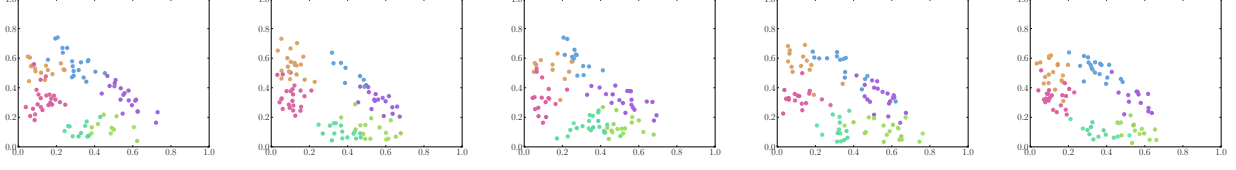


Figure 4: Samples from the symmetrized batch variational Bayes approximate posterior from 5 random restarts. Comparison to Figure 2a shows that symmetrization reintroduces the structure of the true posterior to the approximate posteriors.

representing the overall posterior the best. While the resulting distribution will not be symmetric, it will appear as though it were generated from variational Bayesian inference; this, as mentioned before, is most often fine in practice.

In order to compute the weight of each component, we need to compute its integral over the parameter space. Suppose that each approximate posterior component $q_{\lambda_{ij}}(\theta_j)$ has the following form:

$$q_{\lambda_{ij}}(\theta_j) = h_j(\theta_j) e^{\text{tr}[\lambda_{ij}^T T(\theta_j)] - A_j(\lambda_{ij})} \quad (8)$$

where $h_j(\cdot)$ and $A_j(\cdot)$ are the base measure and log-partition functions for parameter j , respectively. The trace is used in the exponent in case λ_{ij} is specified as a matrix rather than as a single column vector (such as in the example presented in Section 3.1). Thus, given a set of permutation matrices $\{P_i\}_i$, the factor of the weight for the component due to parameter j is

$$W_j(\{P_i\}_i) = \int_{\theta_j} q_{\lambda_{0j}}(\theta_j)^{1-N} \prod_i q_{P_i \lambda_{ij}}(\theta_j). \quad (9)$$

The overall weight of the component is the product over the parameters, so finding the maximum weight component of (7) is equivalent to finding the set of permutation matrices P_i^* that maximizes the product of the W_j ,

$$\{P_i^*\}_i \leftarrow \arg \max_{\{P_i\}_i} \prod_j W_j(\{P_i\}_i). \quad (10)$$

Due to the use of exponential family distributions in the variational approximation, the optimization (10) can be posed as a combinatorial optimization over permutation matrices with a closed-form objective:

$$\begin{aligned} \max_{\{P_i\}_i} \sum_j A_j \left((1-N)\lambda_{0j} + \sum_i P_i \lambda_{ij} \right) \\ \text{s.t. } P_i \in S \quad \forall i \end{aligned} \quad (11)$$

where S is the symmetric group of order equal to the row dimension of the matrices λ_{ij} . Using the convexity of the log-partition function $A_j(\cdot)$, the fact that the objective is affine in its arguments, and the fact that the vertices of the Birkhoff polytope are permutation matrices, one can reformulate (10) as a convex maximization over a polytope:

mulate (10) as a convex maximization over a polytope:

$$\begin{aligned} \max_{\{P_i\}_i} \sum_j A_j \left((1-N)\lambda_{0j} + \sum_i P_i \lambda_{ij} \right) \\ \text{s.t. } P_i^T \mathbf{1} = \mathbf{1}, \quad P_i \mathbf{1} = \mathbf{1}, \quad P_i \geq 0 \quad \forall i \end{aligned} \quad (12)$$

where $\mathbf{1}$ is a vector with all entries equal to 1. Global optimization routines for this problem are intractable for the problem sizes presented by typical Bayesian models (Benson, 1985, Falk and Hoffman, 1986). Thus, the optimization must be solved approximately, where the choice of the approximate method is dependent on the particular form of $A_j(\cdot)$.

As mentioned earlier, this optimization was formulated assuming that all the θ_j were part of a single coupled permutation symmetry set. However, if there are multiple subsets of the parameters $\theta_1, \dots, \theta_K$ that have coupled permutation symmetry, an optimization of the form (12) can be solved for each subset independently. In addition, for any parameter that does not exhibit permutation symmetry, the original naïve posterior merging procedure in (3) may be used. These two statements follow from the exponential family mean field assumption used to construct the individual approximate posteriors q_i .

3 EXPERIMENTS

All experiments were performed on a computer with an Intel Core i7 processor and 12GB of memory.

3.1 DECENTRALIZED MIXTURE MODEL EXAMPLE REVISITED

The AMPS decentralized inference scheme was applied to the Gaussian mixture model example from earlier, with three components having unknown means $\mu = (1.0, -1.0, 3.0)$ and cluster weights $\pi = (0.6, 0.3, 0.1)$, and known variance $\sigma^2 = 0.09$. The prior on each mean μ_i was Gaussian, with mean $\mu_0 = 0.0$ and variance $\sigma_0^2 = 2.0$, while the prior on the weights π was Dirichlet, with parameters $(1.0, 1.0, 1.0)$. The dataset consisting of the same 30 datapoints from the earlier trial was used, where each of 10 learning agents received 3 of the datapoints. Each learning agent used variational Bayesian inference to find their individual posteriors $q_i(\mu, \pi)$, and then used AMPS

to merge them. The only communication required between the agents was a single broadcast of each agent’s individual posterior parameters.

In this example, the AMPS objective¹ was as follows:

$$J_{\text{AMPS}} = -\log \Gamma \left(\sum_{j=1}^3 (\beta_j + 1) \right) + \sum_{j=1}^3 -\frac{\eta_j^2}{4\nu_j} - \frac{1}{2} \log(-2\nu_j) + \log \Gamma(\beta_j + 1) \quad (13)$$

with

$$\begin{aligned} \lambda_{i\mu} &= \begin{bmatrix} \eta_{i1} & \eta_{i2} & \eta_{i3} \\ \nu_{i1} & \nu_{i2} & \nu_{i3} \end{bmatrix}^T, \quad i = 1, \dots, 10 \\ \lambda_{i\pi} &= [\beta_{i1} \quad \beta_{i2} \quad \beta_{i3}]^T, \quad i = 1, \dots, 10 \\ \lambda_{\mu} &= -9\lambda_{0\mu} + \sum_i P_i \lambda_{i\mu} \equiv \begin{bmatrix} \eta_1 & \eta_2 & \eta_3 \\ \nu_1 & \nu_2 & \nu_3 \end{bmatrix}^T \\ \lambda_{\pi} &= -9\lambda_{0\pi} + \sum_i P_i \lambda_{i\pi} \equiv [\beta_1 \quad \beta_2 \quad \beta_3]^T \quad (14) \\ \lambda_{0\mu} &= \begin{bmatrix} 0 & 0 & 0 \\ -0.25 & -0.25 & -0.25 \end{bmatrix}^T \\ \lambda_{0\pi} &= [0 \quad 0 \quad 0]^T \\ \beta_{ij} &= \alpha_{ij} - 1, \quad \eta_{ij} = \frac{\mu_{ij}}{\sigma_{ij}^2}, \quad \nu_{ij} = -\frac{1}{2\sigma_{ij}^2} \end{aligned}$$

where α_{ij} was agent i ’s posterior Dirichlet variational parameter for cluster j , and μ_{ij}/σ_{ij}^2 were agent i ’s posterior normal variational parameter for cluster j . The objective was optimized approximately over the 3×3 permutation matrices P_i by proposing swaps of two rows in P_i , accepting swaps that increased J_{AMPS} , and terminating when no possible swaps increased J_{AMPS} .

The individual posteriors for 3 of the learning agents are shown in Figure 5, while the decentralized posterior over all the agents is shown in Figure 6 alongside its symmetrization (for comparison to the true posterior – this final symmetrization is not required in practice). The AMPS posterior is a much better approximation than the naïve decentralized posterior shown in Figure 2b; this is because the AMPS posterior accounts for parameter permutation symmetry in the model prior to combining the individual posteriors. It may be noted that the decentralized posterior has slightly more uncertainty in it than the batch posterior, but this is to be expected when each learning agent individually receives little information (as demonstrated by the uncertainty in the individual posteriors shown in Figure 5).

¹The AMPS objective for each experiment was constructed using the log-partition function $A_j(\cdot)$ of the relevant exponential family models, which may be found in (Nielsen and Garcia, 2011).

3.2 DECENTRALIZED LATENT DIRICHLET ALLOCATION

The next experiment involved running decentralized variational inference with AMPS on the LDA document clustering model (Blei et al., 2003). The dataset in consideration was the 20 newsgroups dataset, consisting of 18,689 documents with 1,000 held out for testing, and a vocabulary of 11,175 words after removing stop words and stemming the remaining words. Algorithms were evaluated based on their approximate predictive likelihood of 10% of the words in each test document given the remaining 90%, as described in earlier literature (Wang et al., 2011). The variational inference algorithms in this experiment were initialized using smoothed statistics from randomly selected documents.

In LDA, the parameter permutation symmetry lies in the arbitrary ordering of the global word distributions for each topic. In particular, for the 20 newsgroups dataset, decentralized learning agents may learn the 20 Dirichlet distributions with a different ordering; therefore, in order to combine the local posteriors, we use AMPS with the following objective to reorder each agent’s global topics:

$$\begin{aligned} J_{\text{AMPS}} &= \sum_{k=1}^K J_{\text{AMPS},k} = \sum_{k=1}^K \sum_{w=1}^W \log \Gamma(\alpha_{kw}) - \log \Gamma \left(\sum_{w=1}^W \alpha_{kw} \right) \quad (15) \\ \alpha &= (1 - N)\alpha_0 + \sum_{i=1}^N P_i \alpha_i, \quad \alpha, \alpha_i \in \mathbb{R}^{K \times W} \\ \alpha_{0kw} &= \frac{10}{W} \quad K = 20, \quad W = 11,175 \end{aligned}$$

where α_{ikw} is agent i ’s posterior Dirichlet variational parameter for topic k and word w , and the optimization is over $K \times K$ permutation matrices $P_i, i = 1, \dots, N$. For the LDA model, the AMPS objective J_{AMPS} is additive over the topics k ; therefore, J_{AMPS} can be optimized approximately by iteratively solving maximum-weight bipartite matching problems as follows:

1. Initialize the decentralized posterior parameter $\alpha \leftarrow (1 - N)\alpha_0 + \sum_{i=1}^N P_i \alpha_i$ with a set of P_i matrices
2. For each agent i until J_{AMPS} stops increasing:
 - (a) Deassign agent i ’s posterior: $\alpha \leftarrow \alpha - P_i \alpha_i$
 - (b) Form a bipartite graph with decentralized topics k on one side, agent i ’s topics k' on the other, and edge weights $w_{kk'}$ equal to $J_{\text{AMPS},k}$ if agent i ’s topic k' is assigned to the decentralized topic k
 - (c) $P_i \leftarrow$ Maximum weight assignment of agent i ’s topics
 - (d) Reassign agent i ’s posterior: $\alpha \leftarrow \alpha + P_i \alpha_i$

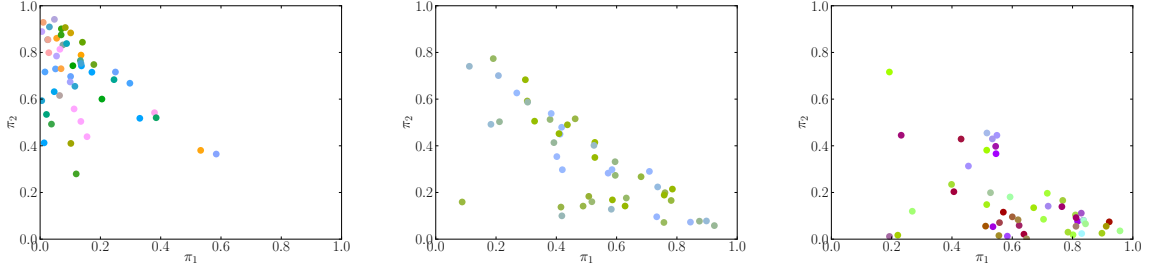


Figure 5: Samples from the individual posterior distributions from variational Bayesian inference for 3 of the learning agents. Note the high level of uncertainty in both the weights (position) and cluster locations (colour) in each posterior.

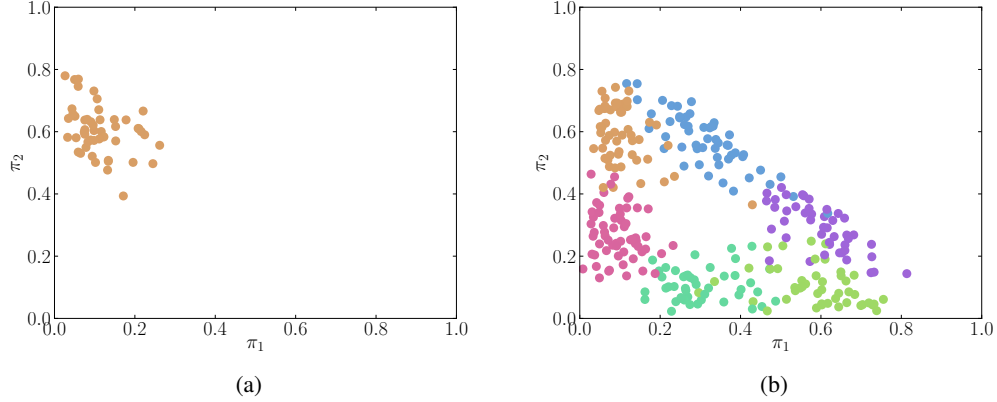


Figure 6: (6a): Samples from the decentralized posterior output by AMPS. Comparison to Figure 5 shows that the AMPS posterior merging procedure improves the posterior possessed by each agent significantly. (6b): Samples from the symmetrized decentralized posterior. This final symmetrization step is not performed in practice; it is simply done here for comparison with Figure 2a.

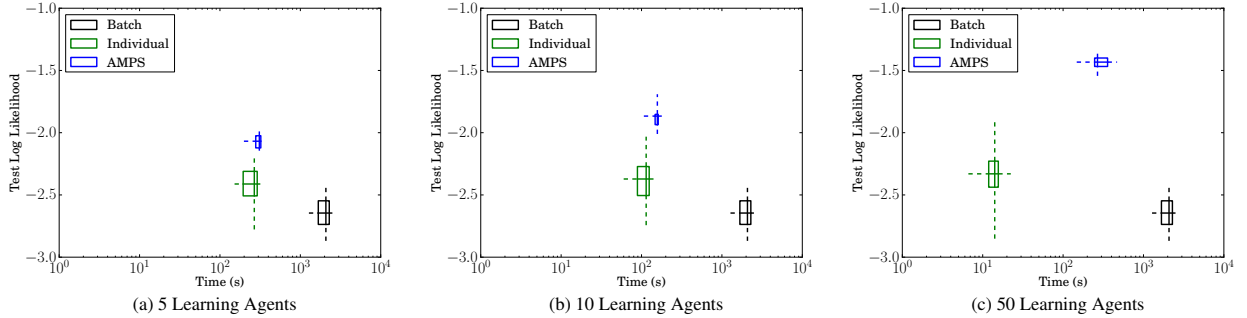


Figure 7: Plots of log likelihood on the test data for 5 (7a), 10 (7b), and 50 (7c) learning agents. The boxes bound the 25th and 75th percentiles of the data with medians shown in the interior, and whiskers show the maximum and minimum values.

First, the performance of decentralized LDA with AMPS was compared to the batch approximate LDA posterior with a varying number of learning agents. Figure 7 shows the test data log likelihood and computation time over 20 trials for the batch posterior, the AMPS decentralized posterior, and each individual agent’s posterior for 5, 10, and 50 learning agents. The results mimic those of the synthetic experiment – the posterior output by AMPS significantly outperforms each individual agent’s posterior, and the effect is magnified as the number of agents increases. Further, there is a much lower variance in the AMPS posterior test log likelihood than for each individual agent. The

batch method tends to get stuck in poor local optima in the variational objective, leading to relatively poor performance, while the decentralized method avoids these pitfalls by solving a number of smaller optimizations and combining the results afterwards with AMPS. Finally, as the number of agents increases, the amount of time required to solve the AMPS optimization increases; reducing this computation time is a potential future goal for research on this inference scheme.

The next test compared the performance of AMPS to SDA-Bayes (Broderick et al., 2013), a recent streaming, dis-

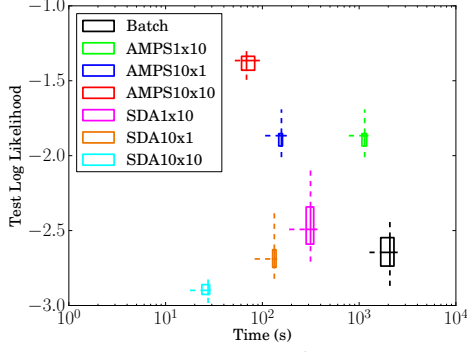


Figure 8: Comparison with SDA-Bayes. The $A \times B$ in the legend names refer to using A learning agents, where each splits their individual batches of data into B subbatches.

tributed variational inference algorithm. The algorithms were tested on 20 trials of each of three settings: one with 1 agent and 10 subbatches of data per agent; one with 10 agents and 1 subbatch of data per agent; and finally, one with 10 agents and 10 subbatches of data per agent. Each agent processed its subbatches in serial. For SDA-Bayes, each agent updated a single distributed posterior after each subbatch. For the decentralized method, each agent used AMPS to combine the posteriors from its own subbatches, and then used AMPS again to combine each agent’s resulting posterior.

Figure 8 shows the results from this procedure. AMPS outperforms SDA-Bayes in terms of test log likelihood, and is competitive in terms of the amount of time it takes to perform inference and then optimize the AMPS objective. This occurs because AMPS takes into account the arbitrary ordering of the topics, while SDA-Bayes ignores this when combining posteriors. An interesting note is that the AMPS10x10 result took less time to compute than the time for 50 agents in Figure 7c, despite the fact that it effectively merged 100 posterior distributions; this hints that developing a hierarchical optimization scheme for AMPS is a good avenue for further exploration. A final note is that using AMPS as described above is not truly a streaming procedure; however, one can rectify this by periodically merging posteriors using AMPS to form the prior for inference on subsequent batches.

3.3 DECENTRALIZED LATENT FEATURE ASSIGNMENT

The last experiment involved running decentralized variational inference with AMPS on a finite latent feature assignment model (Griffiths and Ghahramani, 2005). In this model, a set of K feature vectors $\mu_k \in \mathbb{R}^D$ are sampled from a Gaussian prior $\mu_k \sim \mathcal{N}(0, \sigma_0^2 I)$, and a set of feature inclusion probabilities are sampled from a beta prior $\pi_k \sim \text{Beta}(\alpha_k, \beta_k)$. Finally, for each image i , a set of features z_i are sampled independently from the weights $z_{ik} \sim \text{Be}(\pi_k)$, and the image $y_i \in \mathbb{R}^D$ is sampled from

a Gaussian likelihood $y_i \sim \mathcal{N}(\sum_k \mu_k z_{ik}, \sigma^2 I)$.

Two datasets were used in this experiment. The first was a synthetic dataset with $K = 5$ randomly generated $D = 10$ -dimensional binary feature vectors, feature weights sampled uniformly, and 1300 observations sampled with variance $\sigma^2 = 0.04$, with 300 held out for testing. For this dataset, algorithms were evaluated based on the error between the means of the feature posteriors and the true set of latent features, and based on their approximate predictive likelihoods of a random component in each test observation vector given the other 9 components. The second dataset was a combination of the Yale (Belhumeur et al., 1997) and Caltech² faces datasets, with 581 32×32 frontal images of faces, where 50 were held out for testing. The number of latent features was set to $K = 10$. For this dataset, algorithms were evaluated based on their approximate predictive likelihood of 10% of the pixels in each test image given the remaining 90%, and the inference algorithms were initialized using smoothed statistics from randomly selected images.

The parameter permutation symmetry in the posterior of the latent feature model lies in the ordering of the features μ_k and weights π_k . Therefore, to combine the local posteriors, we use AMPS with the following objective to reorder each agent’s set of features and weights:

$$J_{\text{AMPS}} = \sum_{k=1}^K J_{\text{AMPS},k} = \sum_{k=1}^K \log \Gamma(\alpha_k) + \log \Gamma(\beta_k) - \log \Gamma(\alpha_k + \beta_k) \quad (16)$$

$$- \frac{\eta_k^T \eta_k}{4\nu_k} - \frac{D}{2} \log(-2\nu_k)$$

where $\alpha, \beta \in \mathbb{R}^K$ are the combined posterior beta natural parameters, and $\eta \in \mathbb{R}^{D \times K}$, $\nu \in \mathbb{R}^K$ are the combined posterior normal natural parameters (combined using the $(1 - N) * 0 + \sum_i P_i * i$ rule as described in the foregoing). The priors were $\alpha_{k0} = \beta_{k0} = 1$, $\eta_{k0} = 0 \in \mathbb{R}^D$, and ν_{k0} was estimated from the data. As in the LDA model, the AMPS objective for the latent feature model is additive over the features k ; therefore the optimization was performed using iterative maximum-weight bipartite matchings as described in Section 3.2.

Figure 9 shows the results from the two datasets using batch learning and decentralized learning. For the decentralized results, the posteriors of 5 learning agents were combined using AMPS or the naïve approach (equivalent to SDA5x1 in the notation of Figure 8). Figure 9a shows that AMPS discovers the true set of latent features with a lower 2-norm

²Available online: <http://www.vision.caltech.edu/html-files/archive.html>

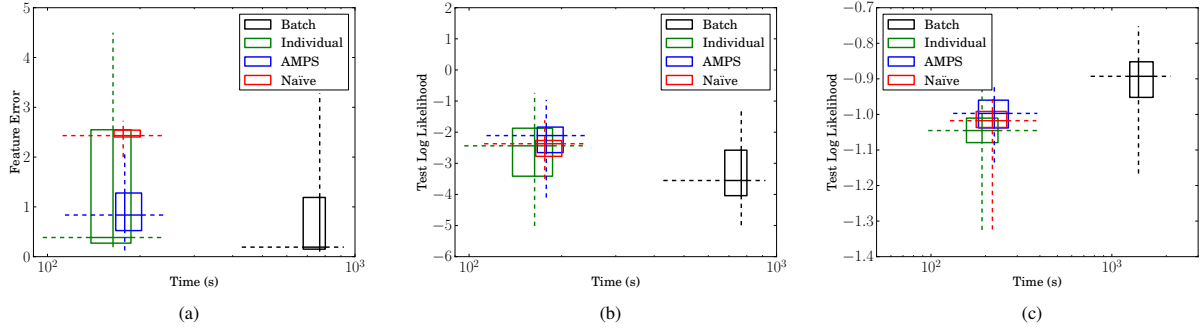


Figure 9: (9a): 2-norm error between the discovered features and the true set for the synthetic dataset. (9b): Test log likelihood for the synthetic dataset. (9c): Test log likelihood on the faces dataset. All distributed/decentralized results were generated using 5 learning agents.

error than both the naïve posterior combination and the individual learning agents, with a comparable error to the batch learning case. However, as shown in Figures 9b (synthetic) and 9c (faces), AMPS only outperforms the naïve approach in terms of predictive log likelihoods on the held-out test set by a small margin. This is due to the flexibility of the latent feature assignment model, in that there are many sets of latent features that explain the observations well.

4 DISCUSSION

This work introduced the Approximate Merging of Posteriors with Symmetry (AMPS) algorithm for approximate decentralized variational inference. AMPS may be used in ad-hoc, asynchronous, and dynamic networks. Experiments demonstrated the modelling and computational advantages of AMPS with respect to batch and distributed learning. Motivated by the examples in Section 3, there is certainly room for improvement of the AMPS algorithm. For example, it may be possible to reduce the computational cost of AMPS by using a hierarchical optimization scheme, rather than the monolithic approach used in most of the examples presented in the foregoing. Further, extending AMPS for use with Bayesian nonparametric models is of interest for cases when the number of latent parameters is unknown a priori, or when there is the possibility that agents learn disparate sets of latent parameters that are not well-combined by optimizing over permutations. Finally, while the approximate optimization algorithms presented herein work well in practice, it would be of interest to find bounds on the performance of such algorithms with respect to the true AMPS optimal solution.

Acknowledgements

This work was supported by the Office of Naval Research under ONR MURI grant N000141110688.

References

- A. Asuncion, P. Smyth, and M. Welling. Asynchronous distributed learning of topic models. In *Advances in Neural Information Processing Systems 21*, 2008.
- P. Belhumeur, J. Hespanha, and D. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Trans. on Pattern Analysis in Machine Intelligence*, 19: 711–720, 1997.
- H. P. Benson. A finite algorithm for concave minimization over a polyhedron. *Naval Research Logistics Quarterly*, 32(1):165–177, 1985.
- D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.
- T. Broderick, N. Boyd, A. Wibisono, A. C. Wilson, and M. I. Jordan. Streaming variational bayes. In *Advances in Neural Information Processing Systems 26*, 2013.
- J. Dean and S. Ghemawat. Mapreduce: Simplified data processing on large clusters. In *6th Symposium on Operating Systems Design and Implementation*, 2004.
- J. E. Falk and K. L. Hoffman. Concave minimization via collapsing polytopes. *Operations Research*, 34(6):919–929, 1986.
- C. S. Fraser, L. F. Bertuccelli, H.-L. Choi, and J. P. How. A hyperparameter consensus method for agreement under uncertainty. *Automatica*, 48:374–380, 2012.
- J. E. Gonzalez, Y. Low, and C. Guestrin. Residual splash for optimally parallelizing belief propagation. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, 2009.
- T. L. Griffiths and Z. Ghahramani. Infinite latent feature models and the indian buffet process. In *Advances in Neural Information Processing Systems 22*, 2005.
- A. Jasra, C. Holmes, and D. Stephens. Markov chain monte carlo methods and the label switching problem in bayesian mixture modeling. *Statistical Science*, 20(1):50–67, 2005.
- D. Lin. Online learning of nonparametric mixture models via sequential variational approximation. In *Advances in Neural Information Processing Systems 26*, 2013.
- D. Newman, A. Asuncion, P. Smyth, and M. Welling. Distributed inference for latent dirichlet allocation. In *Advances in Neural Information Processing Systems 20*, 2007.
- F. Nielsen and V. Garcia. Statistical exponential families: A digest with flash cards. *arXiv:0911.4853v2*, 2011.

- F. Niu, B. Recht, C. Ré, and S. J. Wright. Hogwild!: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems 24*, 2011.
- X. Pan, J. Gonzalez, S. Jegelka, T. Broderick, and M. I. Jordan. Optimistic concurrency control for distributed unsupervised learning. In *Advances in Neural Information Processing Systems 26*, 2013.
- M. A. Paskin and C. E. Guestrin. Robust probabilistic inference in distributed systems. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, 2004.
- M. Rosencrantz, G. Gordon, and S. Thrun. Decentralized sensor fusion with distributed particle filters. In *Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence*, 2003.
- M. Stephens. Dealing with label switching in mixture models. *Journal of the Royal Statistical Society: Series B*, 62(4):795–809, 2000.
- C. Wang, J. Paisley, and D. M. Blei. Online variational inference for the hierarchical dirichlet process. In *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics*, 2011.
- J. Wolfe, A. Haghighi, and D. Klein. Fully distributed EM for very large datasets. In *Proceedings of the 25th International Conference on Machine Learning*, 2008.

Inferring latent structures via information inequalities

R. Chaves^{1*}, L. Luft¹, T. O. Maciel^{1,2}, D. Gross^{1,3}, D. Janzing⁴, B. Schölkopf⁴

¹ Institute for Physics, University of Freiburg, Germany

² Physics Department, Federal University of Minas Gerais, Brazil

³ Freiburg Center for Data Analysis and Modeling, Germany

⁴ Max Planck Institute for Intelligent Systems, Tübingen, Germany

*rafael.chaves@physik.uni-freiburg.de

Abstract

One of the goals of probabilistic inference is to decide whether an empirically observed distribution is compatible with a candidate Bayesian network. However, Bayesian networks with hidden variables give rise to highly non-trivial constraints on the observed distribution. Here, we propose an information-theoretic approach, based on the insight that conditions on *entropies* of Bayesian networks take the form of simple linear inequalities. We describe an algorithm for deriving entropic tests for latent structures. The well-known conditional independence tests appear as a special case. While the approach applies for generic Bayesian networks, we presently adopt the causal view, and show the versatility of the framework by treating several relevant problems from that domain: detecting common ancestors, quantifying the strength of causal influence, and inferring the direction of causation from two-variable marginals.

1 Introduction

Inferring causal relationships from empirical data is one of the prime goals of science. A common scenario reads as follows: Given n random variables X_1, \dots, X_n , infer their causal relations from a list of n -tuples i.i.d. drawn from $P(X_1, \dots, X_n)$. To formalize causal relations, it has become popular to use directed acyclic graphs (DAGs) with random variables as nodes (c.f. Fig. 1) and arrows meaning direct causal influence [23, 28]. Such causal models have been called *causal* Bayesian networks [23], as opposed to traditional Bayesian networks that formalize conditional independence relations without having necessarily a causal interpretation. One of the tasks of causal infer-

ence is to decide which causal Bayesian networks are compatible with empirically observed data.

The most common way to infer the set of possible DAGs from observations is based on the *Markov condition* (c.f. Sect. 2) stating which conditional statistical independencies are implied by the graph structure, and the *faithfulness assumption* stating that the joint distribution is generic for the DAG in the sense that no additional independencies hold [28, 23]. Causal inference via Markov condition and faithfulness has been well-studied for the case where all variables are observable, but some work also refers to latent structures where only a subset is observable [23, 27, 1]. In that case, we are faced with the problem of characterizing the set of *marginal distributions* a given Bayesian network can give rise to. If an observed distribution lies outside the set of marginals of a candidate network, then that model can be rejected as an explanation of the data. Unfortunately, it is widely appreciated that Bayesian networks involving latent variables impose highly non-trivial constraints on the distributions compatible with it [31, 33, 20, 21].

These technical difficulties stem from the fact that the conditional independencies amount to non-trivial algebraic conditions on probabilities. More precisely, the marginal regions are semi-algebraic sets that can, in principle, be characterized by a finite number of polynomial equalities and inequalities [14]. However, it seems that in practice, algebraic statistics is still limited to very simple models.

In order to circumvent this problem, we propose an information-theoretic approach for causal inference. It is based on an entropic framework for treating marginal problems that, perhaps surprisingly, has recently been introduced in the context of Bell's Theorem and the foundations of quantum mechanics [12, 7]. The basic insight is that the *algebraic* condition $p(x, y) = p_1(x)p_2(y)$ for independence becomes a *linear* relation $H(X, Y) = H(X) + H(Y)$ on the level of entropies. This opens up the possibility of us-

ing computational tools such as linear programming to find marginal constraints – which contrasts pleasantly with the complexity of algebraic methods that would otherwise be necessary.

1.1 Results

Our main message is that a significant amount of information about causation is contained in the entropies of observable variables and that there are relatively simple and systematic ways of unlocking that information. We will make that case by discussing a great variety of applications, which we briefly summarize here.

After introducing the geometric and algorithmic framework in Sections 2 & 3, we start with the applications in Section 4.1 which treats instrumentality tests. There, we argue that the non-linear nature of entropy, together with the fact that it is agnostic about the number of outcomes of a random variable, can greatly reduce the complexity of causal tests.

Two points are made in Sec. 4.2, treating an example where the direction of causation between a set of variables is to be inferred. Firstly, that marginal entropies of few variables can carry non-trivial information about conditional independencies encoded in a larger number of variables. This may have practical and statistical advantages. Secondly, we point out applications to tests for quantum non-locality.

In Sec. 4.3 we consider the problem of distinguishing between different hidden common ancestors causal structures. While most of the entropic tests in this paper have been derived using automated linear programming algorithms, this section presents analytic proofs valid for any number of variables.

Finally, Sec. 4.4 details three conceptually important realizations: (1) The framework can be employed to derive quantitative lower bounds on the strength of causation between variables. (2) The degree of violation of entropic inequalities carries an operational meaning. (3) Under some assumptions, we can exhibit novel conditions for distinguishing dependencies created through common ancestors from direct causation.

2 The information-theoretic description of Bayesian networks

In this section we introduce the basic technical concepts that are required to make the present paper self-contained. More details can be found in [23, 12, 7].

2.1 Bayesian networks

Here and in the following, we will consider n jointly distributed discrete random variables (X_1, \dots, X_n) . Uppercase letters label random variables while lowercase label the values taken by these variables, e.g. $p(X_i = x_i, X_j = x_j) \equiv p(x_i, x_j)$.

Choose a *directed acyclic graph* (DAG) which has the X_i 's as its vertices. The X_i 's form a *Bayesian network* with respect to the graph if every variable can be expressed as a function of its parents PA_i and an unobserved noise term N_i , such that the N_i 's are jointly independent. That is the case if and only if the distribution is of the form

$$p(x) = \prod_{i=1}^n p(x_i | pa_i).$$

Importantly, this is equivalent to demanding that the X_i fulfill the *local Markov property*: Every X_i is conditionally independent of its non-descendants ND_i given its parents PA_i : $X_i \perp\!\!\!\perp ND_i | PA_i$.

We allow some of the nodes in the DAG to stand for *hidden variables* that are not directly observable. Thus, the marginal distribution of the observed variables becomes

$$p(v) = \sum_u \prod_{i=1, \dots, m} p(v_i | pa_i) \prod_{j=1, \dots, n-m} p(u_j | pa_j), \quad (1)$$

where $V = (V_1, \dots, V_m)$ are the observable variables and $U = (U_1, \dots, U_{n-m})$ the hidden ones.

2.2 Shannon Entropy cones

Again, we consider a collection of n discrete random variables X_1, \dots, X_n . We denote the set of indices of the random variables by $[n] = \{1, \dots, n\}$ and its power set (i.e., the set of subsets) by $2^{[n]}$. For every subset $S \in 2^{[n]}$ of indices, let X_S be the random vector $(X_i)_{i \in S}$ and denote by $H(S) := H(X_S)$ the associated Shannon entropy given by $H(X_S) = -\sum_{x_S} p(x_S) \log_2 p(x_S)$. With this convention, entropy becomes a function

$$H : 2^{[n]} \rightarrow \mathbb{R}, \quad S \mapsto H(S)$$

on the power set. The linear space of all set functions will be denoted by R_n . For every function $h \in R_n$ and $S \in 2^{[n]}$, we use the notations $h(S)$ and h_S interchangeably.

The region

$$\{h \in R_n \mid h_S = H(S) \text{ for some entropy function } H\}$$

of vectors in R_n that correspond to entropies has been studied extensively in information theory [35]. Its closure is known to be a convex cone, but a tight and

explicit description is unknown. However, there is a standard outer approximation which is the basis of our work: the *Shannon cone* Γ_n . The Shannon cone is the polyhedral closed convex cone of set functions h that respect the following set of linear inequalities:

$$\begin{aligned} h([n] \setminus \{i\}) &\leq h([n]) \\ h(S) + h(S \cup \{i, j\}) &\leq h(S \cup \{i\}) + h(S \cup \{j\}) \\ h(\emptyset) &= 0 \end{aligned} \quad (2)$$

for all $S \subset [n] \setminus \{i, j\}$, $i \neq j$ and $i, j \in [n]$. These inequalities hold for entropy: The first relation – known as *monotonicity* – states that the uncertainty about a set of variables should always be larger than or equal to the uncertainty about any subset of it. The second inequality is the *sub-modularity* condition which is equivalent to the positivity of the conditional mutual information $I(X_i : X_j | X_S) = H(X_{S \cup i}) + H(X_{S \cup j}) - H(X_{S \cup \{i, j\}}) - H(X_S) \geq 0$. The inequalities above are known as the *elementary inequalities* in information theory or the *polymatroidal axioms* in combinatorial optimization. An inequality that follows from the elementary ones is said to be of *Shannon-type*.

The elementary inequalities encode the constraints that the entropies of *any* set of random variables are subject to. If one further demands that the random variables are a Bayesian network with respect to some given DAG, additional relations between their entropies will ensue. Indeed, it is a straight-forward but central realization for the program pursued here, that CI relations faithfully translate to homogeneous linear constraints on entropy:

$$X \perp\!\!\!\perp Y | Z \quad \Leftrightarrow \quad I(X : Y | Z) = 0. \quad (3)$$

The conditional independencies (CI) given by the local Markov condition are sufficient to characterize distributions that form a Bayesian network w.r.t. some fixed DAG. Any such distribution exhibits further CI relations, which can be algorithmically enumerated using the so-called *d-separation criterion* [23]. Let Γ_c be the subspace of R_n defined by the equality (3) for all such conditional independencies. In that language, *the joint distribution of a set of random variables obeys the Markov property w.r.t. to Bayesian network if and only if its entropy vector lies in the polyhedral convex cone $\Gamma_n^c := \Gamma_n \cap \Gamma_c$* , that is, the distribution defines a valid entropy vector (obeying (2)) that is contained in Γ_c . The rest of this paper is concerned with the information that can be extracted from this convex polyhedron.

We remark that this framework can easily be generalized in various directions. E.g., it is simple to incorporate certain quantitative bounds on causal influence. Indeed, small deviations of conditional independence can be expressed as $I(X : Y | Z) \leq \epsilon$ for some

$\epsilon > 0$. This is a (non-homogeneous) linear inequality on R_n . One can add any number of such inequalities to the definition of Γ_n^c while still retaining a convex polyhedron (if no longer a cone). The linear programming algorithm presented below will be equally applicable to these objects. (In contrast to entropies, the set of probability distributions subject to quantitative bounds on various mutual informations seems to be computationally and analytically intractable).

Another generalization would be to replace Shannon entropies by other, non-statistical, information measures. To measure similarities of strings, for instance, one can replace H with Kolmogorov complexity, which (essentially) also satisfies the polymatroidal axioms (2). Then, the conditional mutual information measures conditional algorithmic dependence. Due to the algorithmic Markov condition, postulated in [19], causal structures in nature also imply algorithmic independencies in analogy to the statistical case. We refer the reader to Ref. [30] for further information measures satisfying the polymatroidal axioms.

2.3 Marginal Scenarios

We are mainly interested in situations where not all joint distributions are accessible. Most commonly, this is because the variables X_1, \dots, X_n can be divided into observable ones V_1, \dots, V_m (e.g. medical symptoms) and hidden ones U_1, \dots, U_{n-m} (e.g. putative genetic factors). In that case, it is natural to assume that any subset of observable variables can be *jointly* observed. There are, however, more subtle situations (c.f. Sec. 4.2). In quantum mechanics, e.g., position and momentum of a particle are individually measurable, as is any combination of position and momentum of two distinct particles – however, there is no way to consistently assign a joint distribution to both position and momentum of the same particle [4].

This motivates the following definition: Given a set of variables X_1, \dots, X_n , a *marginal scenario* \mathcal{M} is the collection of those subsets of X_1, \dots, X_n that are assumed to be jointly measurable.

Below, we analyze the Shannon-type inequalities that result from a given Bayesian network and constrain the entropies accessible in a marginal scenario \mathcal{M} .

3 Algorithm for the entropic characterization of any DAG

Given a DAG consisting of n random variables and a marginal scenario \mathcal{M} , the following steps will produce all Shannon-type inequalities for the marginals:

Step 1: *Construct a description of the unconstrained*

Shannon cone. This means enumerating all $n + \binom{n}{2} 2^{n-2}$ elementary inequalities given in (2).

Step 2: Add causal constraints presented as in (3). This corresponds to employing the d -separation criterion to construct all conditional independence relations implied by the DAG.

Step 3: *Marginalization.* Lastly, one has to eliminate all joint entropies not contained in \mathcal{M} .

The first two steps have been described in Sec. 2. We thus briefly discuss the marginalization, first from a geometric, then from an algorithmic perspective.

Given a set function $h : 2^{[n]} \rightarrow \mathbb{R}$, its restriction $h|_{\mathcal{M}} : \mathcal{M} \rightarrow \mathbb{R}$ is trivial to compute: If h is expressed as a vector in R_n , we just drop all coordinates of h which are indexed by sets outside of \mathcal{M} . Geometrically, this amounts to a projection $P_{\mathcal{M}} : \mathbb{R}^{2^n} \rightarrow \mathbb{R}^{|\mathcal{M}|}$. The image of the constrained cone Γ_n^c under the projection $P_{\mathcal{M}}$ is again a convex cone, which we will refer to as $\Gamma^{\mathcal{M}}$. Recall that there are two dual ways of representing a polyhedral convex cone: in terms of either its extremal rays, or in terms of the inequalities describing its facets [2]. To determine the projection $\Gamma^{\mathcal{M}}$, a natural possibility would be to calculate the extremal rays of Γ_n^c and remove the irrelevant coordinates of each of them. This would result in a set of rays generating $\Gamma^{\mathcal{M}}$. However, Steps 1 & 2 above give a representation of Γ_n^c in terms of inequalities. Also, in order to obtain readily applicable tests, we would prefer an inequality presentation of $\Gamma^{\mathcal{M}}$. Thus, we have chosen an algorithmically more direct (if geometrically more opaque) procedure by employing Fourier-Motzkin elimination – a standard linear programming algorithm for eliminating variables from systems of inequalities [34].

In the remainder of the paper, we will discuss applications of inequalities resulting from this procedure to causal inference.

4 Applications

4.1 Conditions for Instrumentality

An instrument Z is a random variable that under certain assumptions helps identifying the causal effect of a variable X on another variable Y [16, 22, 5]. The simplest example is given by the instrumentality DAG in Fig. 1 (a), where Z is an instrumental variable and the following independencies are implied: (i) $I(Z : Y|X, U) = 0$ and (ii) $I(Z : U) = 0$. The variable U represents all possible factors (observed and unobserved) that may effect X and Y . Because conditions (i) and (ii) involve an unobservable variable U , the use of an instrument Z can only be justified if the observed

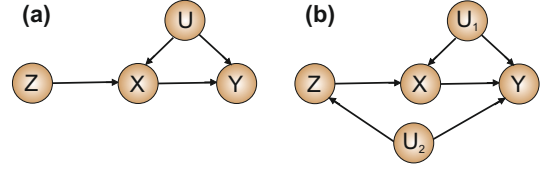


Figure 1: DAG (a) represents the instrumental scenario. DAG (b) allows for a common ancestor between Z and Y : unless some extra constraint is imposed (e.g. $I(Y, U_2) \leq \epsilon$) this DAG is compatible with any probability distribution for the variables X , Y and Z .

distribution falls inside the compatibility region implied by the instrumentality DAG. The distributions compatible with this scenario can be written as

$$p(x, y|z) = \sum_u p(u)p(y|x, u)p(x|z, u) \quad (4)$$

Note that (4) can be seen as a convex combination of deterministic functions assigning the values of X and Y [22, 5, 25]. Thus, the region of compatibility associated with $p(x, y|z)$ is a polytope and all the probability inequalities characterizing it can in principle be determined using linear programming. However, as the number of values taken by the variables increases, this approach becomes intractable [5] (see below for further comments). Moreover, if we allow for variations in the causal relations, e.g. the one shown in DAG (b) of Fig. 1, the compatibility region is not a polytope anymore and computationally challenging algebraic methods would have to be used [15]. For instance, the quantifier elimination method in [15] is unable to deal with the instrumentality DAG even in the simplest case of binary variables. We will show next how our framework can easily circumvent such problems.

Proceeding with the algorithm described in Sec. 3, one can see that after marginalizing over the latent variable U , the only non-trivial entropic inequality constraining the instrumental scenario is given by

$$I(Y : Z|X) + I(X : Z) \leq H(X). \quad (5)$$

By “non-trivial”, we mean that (5) is not implied by monotonicity and sub-modularity for the observable variables. The causal interpretation of (5) can be stated as follows: Since Z influence Y only through X , if the dependency between X and Z is large, then necessarily the dependency between Y and Z conditioned on knowing X should be small.

We highlight the fact that, irrespective of how many values the variables X , Y and Z may take (as long as they are discrete), (5) is the only non-trivial entropic

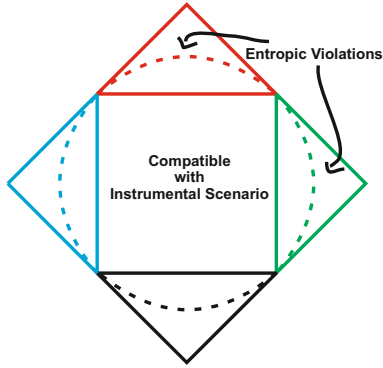


Figure 2: A comparison between the entropic and the probabilistic approach. The squares represent the polytope of distributions compatible with the instrumental DAG. Each facet in the square corresponds to one of the 4 non-trivial inequalities valid for binary variables [22, 5]. The triangles over the squares represent probability distributions that fail to be compatible with the instrumental constraints. Distributions outside the dashed curve are detected by the entropic inequality (5). Due to its non-linearity in terms of probabilities, (5) detects the non-compatibility associated with different probability inequalities. See [8] for more details.

constraint bounding the distributions compatible with the instrumentality test. This is in stark contrast with the probabilistic approach, for which the number of linear inequalities increases exponentially with the number of outcomes of the variables [5]. There is, of course, a price to pay for this concise description: There are distributions that are not compatible with the instrumental constraints, but fail to violate (5). In this sense, an entropic inequality is a necessary but not sufficient criterion for compatibility. However, it is still surprising that a single entropic inequality can carry information about causation that is in principle contained only in exponentially many probabilistic ones. This effect stems from the non-linear nature of entropy¹ and is illustrated in Fig. 2.

Assume now that some given distribution $p(x, y|z)$ is incompatible with the instrumental DAG. That could be due to some dependencies between Y and Z mediated by a common hidden variable U_2 as shown in DAG (b) of Fig. 1. Clearly, this DAG can explain any

¹We remark that the reduction of descriptive complexity resulting from the use of non-linear inequalities occurs for other convex bodies as well. The simplest example along these lines is the Euclidean unit ball B . It requires infinitely many linear inequalities to be defined (namely $B = \{x \mid (x, y) \leq 1 \forall y, \|y\|_2 \leq 1\}$). These can, of course, all be subsumed by the single non-linear condition $\|x\|_2 \leq 1$.

distribution $p(x, y|z)$ and therefore is not very informative. Notwithstanding, with our approach we can for instance put a quantitative lower bound on how dependent Y and U_2 need to be. Following the algorithm in Sec. 3, one can see that the only non-trivial constraint on the dependency between Y and U_2 is given by $I(Y : U_2) \leq H(Y|X)$. This inequality imposes a kind of *monogamy of correlations*: if the uncertainty about Y is small given X , their dependency is large, implying that Y is only slightly correlated with U_2 , since the latter is statistically independent of X .

4.2 Inferring direction of causation

As mentioned before, if all variables in the DAG are observed, the conditional independencies implied by the graphical model completely characterize the possible probability distributions [24]. For example, the DAGs displayed in Fig. 3 display a different set of CIs. For both DAGs we have $I(X : Z|Y, W) = 0$, however for DAG (a), it holds that $I(Y : W|X) = 0$ while for DAG (b) $I(Y : W|Z) = 0$. Hence, if the joint distributions of (Y, W, X) and (Y, W, Z) are accessible, then CI information can distinguish between the two networks and thus reveal the “direction of causation”.

In this section, we will show that the same is possible even if only two variables are jointly accessible at any time. We feel this is relevant for three reasons.

First – and somewhat subjectively – we believe the insight to be interesting from a fundamental point of view. Inferring the direction of causation between two variables is a notoriously thorny issue, hence it is far from trivial that it can be done from information about several pairwise distributions.

The second reason is that there are situations where joint distributions of many variables are unavailable due to practical or fundamental reasons. We have already mentioned quantum mechanics as one such example – and indeed, the present DAGs can be related to tests for quantum non-locality. We will briefly discuss the details below. But also purely classical situations are conceivable. For instance, Mendelian randomization is a good example where the joint distribution on all variables is often unavailable [10].

Thirdly, the “smoothing effect” of marginalizing may simplify the statistical analysis when only few samples are available. Conditioning on many variables or on variables that attain many different values often amounts to conditioning on events that happened only once. Common χ^2 -tests for CI [32] involve divisions by empirical estimates of variance, which lead to nonsensical results if no variance is observed. Testing for CI in those situations requires strong assumptions (like smoothness of dependencies) and remains

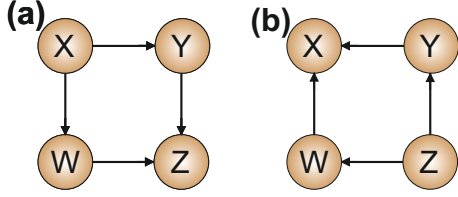


Figure 3: DAGs with no hidden variables and opposite causation directions. The DAGs can be distinguished based on the CIs induced by them. However, if only pairwise information is available one must resort to the marginalization procedure described in Sec. 3.

a challenging research topic [13, 36]. Two-variable marginals, while containing strictly less information than three-variable ones, show less fluctuations and might thus be practically easier to handle. This benefit may not sound spectacular as long as it refers to 2- versus 3-variable marginals. However, in general, our formalism can provide inequality constraints for k -variable marginals from equality constraints that involve ℓ -variable marginals for $\ell \gg k$.

We note that causal inference schemes using only pairwise mutual information is already known for trees, i.e., DAGs containing no undirected cycles. The data processing inequality implies that for every node, the mutual information to a direct neighbor cannot be smaller than the one with the neighbor of this neighbor. Hence one can find adjacencies based on pairwise mutual information only. This has been used e.g. for phylogenetic trees [17, 9]. In that sense, our results generalize these ideas to DAGs with cycles.

The non-trivial constraints on two-variable entropies given by our algorithm for the DAG (a) of Fig. 3 are:

$$\begin{aligned}
 H_Y - H_X - H_{YW} + H_{XW} &\leq 0 \\
 H_W - H_X - H_{YW} + H_{XY} &\leq 0 \\
 H_{WZ} - H_{YW} - H_{XZ} + H_{XY} &\leq 0 \\
 H_{YZ} - H_{YW} - H_{XZ} + H_{XW} &\leq 0 \\
 H_Y - H_X + H_W - H_{WZ} - H_{YZ} + H_{XZ} &\leq 0 \\
 H_Z - H_X - H_{YW} - H_{XZ} + H_{XW} + H_{XY} &\leq 0 \\
 H_Z + H_X \\
 + H_{YW} + H_{XZ} - H_{XW} - H_{XY} - H_{WZ} - H_{YZ} &\leq 0.
 \end{aligned} \tag{6}$$

The ones for DAG (b) are obtained by the substitution $X \leftrightarrow Z$. Invariant under this, the final inequality is valid for both scenarios. In contrast, the first six inequalities can be used to distinguish the DAGs.

As an example, one can consider the following structural equations compatible only with the DAG (b): Z is a uniformly distributed m -valued random variable,

$Y = W = Z$, and $X = Y \oplus W$ (addition modulo m). A direct calculation shows that the first inequality in (6) is violated, thus allowing one to infer the correct direction of the arrows in the DAG.

As alluded to before, we close this section by mentioning a connection to quantum non-locality [4]. Using the linear programming algorithm, one finds that the final inequality in (6) is actually valid for *any* distribution of four random variables, not only those that constitute Bayesian networks w.r.t. the DAGs in Fig. 3. In that sense it seems redundant, or, at best, a sanity check for consistency of data. It turns out, however, that it can be put to non-trivial use. While the purpose of causal inference is to check compatibility of data with a presumed causal structure, the task of quantum non-locality is to devise tests of compatibility with classical probability theory as a whole. Thus, if said inequality is violated in a quantum experiment, it follows that there is no way to construct a joint distribution of all four variables that is consistent with the observed two-variable marginals – and therefore that classical concepts are insufficient to explain the experiment.

While not every inequality which is valid for all classical distributions can be violated in quantum experiments, the constraints in (6) do give rise to tests with that property. To see this, we further marginalize over $H(X, Z)$ and $H(Y, W)$ to obtain

$$H_{XY} + H_{XW} + H_{YZ} - H_{WZ} - H_Y - H_X \leq 0 \tag{7}$$

(and permutations thereof). These relations have been studied as the “entropic version of the CHSH Bell inequality” in the physics literature [6, 12, 7], where it is shown that (7) can be employed to witness that certain measurements on quantum systems do not allow for a classical model.

4.3 Inference of common ancestors in semi-Markovian models

In this section, we re-visit in greater generality the problem considered in [29]: using entropic conditions to distinguish between hidden common ancestors.

Any distribution of a set of n random variables can be achieved if there is one latent parent (or *ancestor*) common to all of them [23]. However, if the dependencies can also be obtained from a less expressive DAG – e.g. one where at most two of the observed variables share an ancestor – then Occam’s Razor would suggest that this model is preferable. The question is then: what is the simplest common ancestor causal structure explaining a given set of observations?

One should note that unless we are able to intervene in the system under investigation, in general it may

be not possible to distinguish direct causation from a common cause. For instance, consider the DAGs (a) and (c) displayed in Fig. 4. Both DAGs are compatible with any distribution and thus it is not possible to distinguish between them from passive observations alone. For this reason and also for simplicity, we restrict our attention to semi-Markovian models where all the observable variables are assumed to have no direct causation on each other or on the hidden variables. Also, the hidden variables are assumed to be mutually independent. It is clear then that all dependencies between the observed quantities can only be mediated by their hidden common ancestors. We refer to such models as common ancestors (CM) DAGs. We reinforce, however, that our framework can also be applied in the most general case. As will be explained in more details in Sec. 4.4, in some cases, common causes can be distinguished from direct causation. Our framework can also be readily applied in these situations.

We begin by considering the simplest non-trivial case, consisting of three observed variables [29, 11, 7]. If no conditional independencies between the variables occur, then the graphs in Fig. 4 (a) and (b) represent the only compatible CM DAGs. Applying the algorithm described in Sec. 3 to the model (b), we find that one non-trivial class of constraints is given by

$$I(V_1 : V_2) + I(V_1 : V_3) \leq H(V_1) \quad (8)$$

and permutations thereof [11, 7].

It is instructive to pause and interpret (8). It states, for example, that if the dependency between V_1 and V_2 is maximal ($I(V_1 : V_2) = H(V_1)$) then there should be no dependency at all between V_1 and V_3 ($I(V_1 : V_3) = 0$). Note that $I(V_1 : V_2) = H(V_1)$ is only possible if V_1 is a deterministic function of the common ancestor U_{12} alone. But if V_1 is independent of U_{13} , it cannot depend on V_3 and thus $I(V_1 : V_3) = 0$.

Consider for instance a distribution given by

$$p(v_1, v_2, v_3) = \begin{cases} 1/2 & , \text{ if } v_1 = v_2 = v_3 \\ 0 & , \text{ otherwise} \end{cases} \quad (9)$$

This stands for a perfect correlation between all the three variables and clearly cannot be obtained by pairwise common ancestors. This incompatibility is detected by the violation of (8).

We now establish the following generalization of (8) to an arbitrary number of observables:

Theorem 1 *For any distribution that can be explained by a CM DAG where each of the latent ancestors influences at most m of the observed variables,*

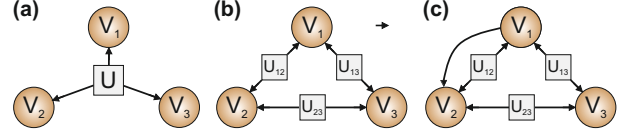


Figure 4: Models (a) and (b) are CM DAGs for three observable variables V_1, V_2, V_3 . Unlike (b), DAG (a) is compatible with any observable distribution. DAG (c) involves a direct causal influence between the observable variable V_1 and V_2 .

we have

$$\sum_{\substack{i=1, \dots, n \\ i \neq j}} I(V_i : V_j) \leq (m-1)H(V_j). \quad (10)$$

We present the proof for the case $m = 2$ while the general proof can be found in the supplemental material.

Lemma 1 *In the setting of Thm. 1 for $m = 2$:*

$$\sum_{i=2}^n H(V_j U_{ji}) \geq (N-2)H(V_j) + H(V_j \bigcup_{i=2}^N U_{ji}). \quad (11)$$

Proof. (By induction) We treat the case $j = 1$ w.l.o.g. For $n = 2$ equality holds trivially. Now assuming the validity of the inequality for any n :

$$\sum_{i=2}^{n+1} H(V_1 U_{1i}) \geq (n-2)H(V_1) \quad (12)$$

$$+ H(V_1 \bigcup_{i=2}^n U_{1i}) + H(V_1 U_{1(n+1)}) \\ \geq [(n+1)-2]H(V_1) + H(V_1 \bigcup_{i=2}^{n+1} U_{1i}). \quad (13)$$

From (12) to (13) we have used sub-modularity. \square

Proof of Theorem 1. Apply the data processing inequality to the left-hand side of (10) to obtain

$$\sum_{i=2}^n I(A_1 : A_i) \leq \sum_{i=2}^n I(A_1 : U_{1i}) \\ = (n-1)H(A_1) + \sum_{i=2}^n H(\lambda_{1i}) - \sum_{i=2}^n H(A_1 \lambda_{1i}).$$

With Lemma 1, we get

$$\sum_{i=2}^n I(V_1 : V_i) \leq (n-1)H(V_1) + \sum_{i=2}^n H(U_{1i}) \\ - [(n-2)H(V_1) + H(V_1 \bigcup_{i=2}^n U_{1i})].$$

The mutual independence of hidden variables yields $\sum_{i=2}^n H(U_{1i}) = H(\bigcup_{i=2}^n U_{1i})$ implying that

$$\sum_{i=2}^n I(V_1 : V_i) \leq H(V_1) - H(V_1 | \bigcup_{i=2}^n U_{1i}) \leq H(V_1).$$

\square

We highlight the fact that Ineq. (10) involves only pairwise distributions – the discussion in Sec. 4.2 applies. Following our approach, one can derive further entropic inequalities, in particular involving the joint entropy of all observed variables. A more complete theory will be presented elsewhere.

4.4 Quantifying causal influences

Unlike conditional independence, mutual information captures dependencies in a quantitative way. In this section, we show that our framework allows one to derive non-trivial bounds on the strength of causal links. We then go on to present two corollaries of this result: First, it follows that the degree of violation of an entropic inequality often carries an operational meaning. Second, under some assumptions, the finding will allow us to introduce a novel way of distinguishing dependence created through common ancestors from direct causal influence.

Various measures of causal influence have been studied in the literature. Of particular interest to us is the one recently introduced in [18]. The main idea is that the causal strength $\mathcal{C}_{X \rightarrow Y}$ between a variable X on another variable Y should measure the impact of an intervention that removes the arrow between them. Ref. [18] draws up a list of reasonable postulates that a measure of causal strength should fulfill. Of special relevance to our information-theoretic framework is the axiom stating that

$$\mathcal{C}_{X \rightarrow Y} \geq I(X : Y | \text{PA}_Y^X), \quad (14)$$

where PA_Y^X stands for the parents of variable Y other than X . We focus on this property, as the quantity $I(X : Y | \text{PA}_Y^X)$ appears naturally in our description and thus allows us to bound any measure of causal strength $\mathcal{C}_{X \rightarrow Y}$ for which (14) is valid.

To see how this works in practice, we start by augmenting the common ancestor scenario considered in the previous section. Assume that now we do allow for direct causal influence between two variables, in addition to pairwise common ancestors – c.f. Fig. 4 (c). Then (14) becomes $\mathcal{C}_{V_1 \rightarrow V_2} \geq I(V_1 : V_2 | U_{12}, U_{13})$. We thus re-run our algorithm, this time with the unobservable quantity $I(V_1 : V_2 | U_{12}, U_{13})$ included in the marginal scenario. The result is

$$I(V_1 : V_2 | U_{12}, U_{13}) \geq I(V_1 : V_2) + I(V_1 : V_3) - H(V_1), \quad (15)$$

which lower-bounds the causal strength in terms of observable entropies.

The same method yields a particularly concise and relevant result when applied to the instrumental test of Sec. 4.1. The instrumental DAG may stand, for example, for a clinical study about the efficacy of some

drug where Z would label the treatment assigned, X the treatment received, Y the observed response and U for any observed or unobserved factors affecting X and Y . In this case we would be interested not only in checking the compatibility with the presumed causal relations but also the direct causal influence of the drug on the expected observed response, that is, $\mathcal{C}_{X \rightarrow Y}$. After the proper marginalization we conclude that $\mathcal{C}_{X \rightarrow Y} \geq I(Y : Z) - H(X)$, a strikingly simple, but non-trivial bound that can be computed from the observed quantities alone. Likewise, if one allows the instrumental DAG to have an arrow connecting Z and Y , one finds

$$\mathcal{C}_{Z \rightarrow Y} \geq I(Y : Z | X) + I(X : Z) - H(X). \quad (16)$$

The findings presented here can be re-interpreted in two ways:

First, note that the right hand side of the lower bound (15) is nothing but Ineq. (8), a constraint on distributions compatible with DAG 3 (b). Similarly, the r.h.s. of (16) is just the degree of violation of the entropic instrumental inequality (5).

We thus arrive at the conceptually important realization that the entropic conditions proposed here offer more than just binary tests. To the contrary, their degree of violation is seen to carry a quantitative meaning in terms of strengths of causal influence.

Second, one can interpret the results of this sections as providing a novel way to distinguish between DAGs (a) and (c) in Fig. 4 without experimental data. Assume that we have some information about the physical process that could facilitate direct causal influence from V_1 to V_2 in (c), and that we can use that prior information to put a quantitative upper bound on $\mathcal{C}_{V_1 \rightarrow V_2}$. Then we must reject the direct causation model (c) in favor of a common ancestor explanation (a), as soon as the observed dependencies violate the bound (15). As an illustration, the perfect correlations exhibited by the distribution (9) is incompatible with DAG (c), as long as $\mathcal{C}_{V_1 \rightarrow V_2}$ is known to be smaller than 1.

5 Statistical Tests

In this section, we briefly make the point that inequality-based criteria immediately suggest test statistics which can be used for testing hypotheses about causal structures. While a thorough treatment of statistical issues is the subject of ongoing research [3, 26], it should become plain that the framework allows to derive non-trivial tests in a simple way.

Consider an inequality $I := \sum_{S \subset 2^{[n]}} c_S H(S) \leq 0$ for suitable coefficients c_S . Natural candidates for test

statistics derived from it would be $T_I := \sum_S c_S \hat{H}(S)$ or $T'_I := \frac{T_I}{\sqrt{\text{var}(T_I)}}$, where $\hat{H}(S)$ is the entropy of the empirical distribution of X_S , and var is some consistent estimator of variance (e.g. a bootstrap estimator). If the inequality I is fulfilled for some DAG G , then a test with null hypothesis “data is compatible with G ” can be designed by testing $T_I \leq t$ or $T'_I \leq t$, for some critical value $t > 0$. In an asymptotic regime, there could be reasonable hope to analytically characterize the distribution of T'_I . However, in the more relevant small sample regime, one will probably have to resort to Monte Carlo simulations in order to determine t for a desired confidence level. In that case, we prefer to use T_I , by virtue of being “less non-linear” in the data.

We have performed a preliminary numerical study using the DAG given in Fig. 4 (b) together with Ineq. (8). We have simulated experiments that draw 50 samples from various distributions of three binary random variables V_1, V_2, V_3 and compute the test statistic T_I . To test at the 5%-level, we must choose t large enough such that for all distributions p compatible with 4(b), we have a type-I error rate $\Pr_p[T_I > t]$ below 5%. We have employed the following heuristics for finding t : (1) It is plausible that the highest type-I error rate occurs for distributions p that reach equality $\mathbb{E}_p[\hat{I}] = 0$; (2) This occurs only if V_1 is a deterministic function of V_2 and V_3 . From there, it follows that V_1 must be a function of one of V_2 or V_3 and we have used a Monte Carlo simulation with (V_2, V_3) uniformly random and $V_1 = V_2$ to find $t = .0578$. Numerical checks failed to identify distributions with higher type-I rate (though we have no proof). Fig. 5 illustrates the resulting test.

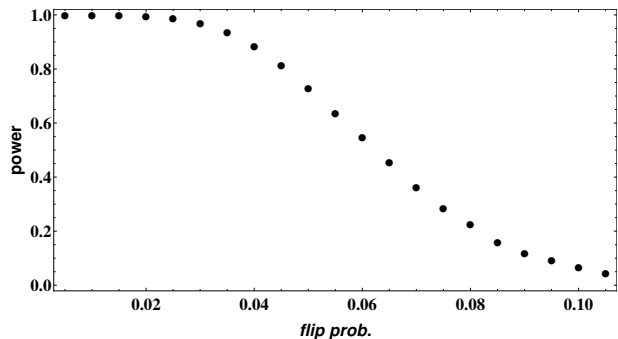


Figure 5: Power (1 minus type-II error) of the test $T_I \geq t$ for the DAG Fig. 4(b) derived from Ineq. (8) using 50 samples. The test was run on a distribution obtained by starting with three perfectly correlated binary random variables as in (9) and then inverting each of the variables independently with a given “flip probability” (x axis). Every data point is the result of 10000 Monte Carlo simulations.

6 Conclusions

Hidden variables imply nontrivial constraints on observable distributions. While we cannot give a complete characterization of these constraints, we show that a number of nontrivial constraints can be elegantly formulated in terms of entropies of subsets of variables. These constraints are linear (in)equalities, which lend themselves well to algorithmic implementation.

Remarkably, our approach only requires the polymatroidal axioms, and thus also applies to various information measures other than Shannon entropy. Some of these may well be relevant to causal inference and structure learning and may constitute an interesting topic for future research.

Acknowledgements

We acknowledge support by the Excellence Initiative of the German Federal and State Governments (Grant ZUK 43), the Research Innovation Fund from the University of Freiburg and the Brazilian research agency CNPq. DG’s research is supported by the US Army Research Office under contracts W911NF-14-1-0098 and W911NF-14-1-0133 (Quantum Characterization, Verification, and Validation).

References

- [1] R.A. Ali, T. Richardson, P. Spirtes, and J. Zhang. Orientation rules for constructing Markov equivalence classes for maximal ancestral graphs. Technical Report TR 476, University of Washington, 2005.
- [2] C. D. Aliprantis and R. Tourky. *Cones and duality*. American Mathematical Soc., 2007.
- [3] F. Bartolucci and A. Forcina. A likelihood ratio test for mtp2 within binary variables. *Annals of Statistics*, pages 1206–1218, 2000.
- [4] J. S. Bell. On the Einstein–Podolsky–Rosen paradox. *Physics*, 1:195, 1964.
- [5] B. Bonet. Instrumentality tests revisited. *UAI 2001*, pages 48–55.
- [6] S. L. Braunstein and C. M. Caves. Information-theoretic Bell inequalities. *Phys. Rev. Lett.*, 61(6):662–665, 1988.
- [7] R. Chaves, L. Luft, and D. Gross. Causal structures from entropic information: Geometry and novel scenarios. *New J. Phys.*, 16:043001, 2014.

- [8] R. Chaves. Entropic inequalities as a necessary and sufficient condition to noncontextuality and locality. *Phys. Rev. A*, 87:022102, 2013.
- [9] X. Chen, X. Kwong, and M. Li. A compression algorithm for DNA sequences and its applications in genome comparison. In *Proc. of the 4th Annual Int. Conf. on Computational Molecular Biology*, page 107, 2000.
- [10] V. Didelez and N. Sheehan. Mendelian randomization as an instrumental variable approach to causal inference. *Stat Meth Med Res*, 16(4):309–330, 2007.
- [11] T. Fritz. Beyond Bell’s theorem: correlation scenarios. *New J. Phys.*, 14:103001, 2012.
- [12] T. Fritz and R. Chaves. Entropic inequalities and marginal problems. *IEEE Transact. on Inf. Theory*, 59:803, 2013.
- [13] K. Fukumizu, A. Gretton, X. Sun, and B. Schölkopf. Kernel measures of conditional dependence. *NIPS 2008*, 21:489–496.
- [14] D. Geiger and C. Meek. Graphical models and exponential families. *UAI 1998*, pages 156–165.
- [15] D. Geiger and C. Meek. Quantifier elimination for statistical problems. *UAI 1999*, pages 226–235.
- [16] A. S. Goldberger. Structural equation methods in the social sciences. *Econometrica*, 40(6):pp. 979–1001, 1972.
- [17] S. Grumbach and F. Tahi. A new challenge for compression algorithms: genetic sequences. *Information Processing & Management*, 30(6):875–886, 1994.
- [18] D. Janzing, D. Balduzzi, M. Grosse-Wentrup, and B. Schölkopf. Quantifying causal influences. *Annals of Statistics*, 41(5):2324–2358, 10 2013.
- [19] D. Janzing and B. Schölkopf. Causal inference using the algorithmic markov condition. *IEEE Transactions on Information Theory*, 56(10):5168–5194, 2010.
- [20] C. Kang and J. Tian. Inequality constraints in causal models with hidden variables. *UAI 2006*, pages 233–240.
- [21] C. Kang and J. Tian. Polynomial constraints in causal Bayesian networks. *UAI 2007*, pages 200–208.
- [22] J. Pearl. On the testability of causal models with latent and instrumental variables. *UAI 1995*, pages 435–443.
- [23] J. Pearl. *Causality*. Cambridge University Press, 2009.
- [24] J. Pearl, D. Geiger, and T. Verma. *The logic of influence diagrams*. In *Influence Diagrams, Belief Nets and Decision Analysis*. JohnWiley and Sons, Inc., NY, 1990.
- [25] Roland R Ramsahai. Causal bounds and observable constraints for non-deterministic models. *The Journal of Machine Learning Research*, 13:829–848, 2012.
- [26] RR Ramsahai and SL Lauritzen. Likelihood analysis of the binary instrumental variable model. *Biometrika*, 98(4):987–994, 2011.
- [27] T. Richardson and P. Spirtes. Ancestral graph Markov models. *Annals of Statistics*, 30(4):962–1030, 2002.
- [28] P. Spirtes, N. Glymour, and R. Scheienes. *Causation, Prediction, and Search, 2nd ed.* MIT Press, 2001.
- [29] B. Steudel and N. Ay. Information-theoretic inference of common ancestors. *arXiv:1010.5720*, 2010.
- [30] B. Steudel, D. Janzing, and B. Schölkopf. Causal markov condition for submodular information measures. *COLT 2010*. pages 464–476.
- [31] J. Tian and J. Pearl. On the testable implications of causal models with hidden variables. *UAI 2002*, pages 519–527.
- [32] G. J. G. Upton. Conditional independence, the mantel-haenszel test, and the yates correction. *The American Statistician*, 54(2):112–115, 2000.
- [33] G. Ver Steeg and A. Galstyan. A sequence of relaxations constraining hidden variable models. *UAI 2011*, pages 717–727.
- [34] H. P. Williams. Fourier’s method of linear programming and its dual. *Amer. Math. Monthly*, 93(9):681–695, 1986.
- [35] R. W. Yeung. *Information theory and network coding*. Information technology–transmission, processing, and storage. Springer, 2008.
- [36] K. Zhang, P. Peters, D. Janzing, and B. Schölkopf. Kernel-based conditional independence test and application in causal discovery. *UAI 2011*, pages 804–813.

Near-optimal Adaptive Pool-based Active Learning with General Loss

Nguyen Viet Cuong

Department of Computer Science
National University of Singapore
nvcuong@comp.nus.edu.sg

Wee Sun Lee

Department of Computer Science
National University of Singapore
leews@comp.nus.edu.sg

Nan Ye

Department of Computer Science
National University of Singapore
yen@comp.nus.edu.sg

Abstract

We consider adaptive pool-based active learning in a Bayesian setting. We first analyze two commonly used greedy active learning criteria: the maximum entropy criterion, which selects the example with the highest entropy, and the least confidence criterion, which selects the example whose most probable label has the least probability value. We show that unlike the non-adaptive case, the maximum entropy criterion is not able to achieve an approximation that is within a constant factor of optimal policy entropy. For the least confidence criterion, we show that it is able to achieve a constant factor approximation to the optimal version space reduction in a worst-case setting, where the probability of labelings that have not been eliminated is considered as the version space. We consider a third greedy active learning criterion, the Gibbs error criterion, and generalize it to handle arbitrary loss functions between labelings. We analyze the properties of the generalization and its variants, and show that they perform well in practice.

1 INTRODUCTION

We study pool-based active learning (McCallum and Nigam, 1998) where the training data are sequentially selected and labeled from a pool of unlabeled examples, with the aim of having good performance after only a small number of examples are labeled. In practice, the selection of the next example to be labeled is usually done by greedy optimization of some appropriate objective function.

In this paper, we consider adaptive algorithms for pool-based active learning with a budget of k queries in a Bayesian setting. We examine three commonly used greedy criteria and their performance guarantees. We also generalize one of the criteria, study its properties and show that it performs well in practice.

One of the most commonly used criteria is the *maximum entropy criterion*: select the example with maximum label entropy given the observed labels (Settles, 2010). In the non-adaptive case where the set of examples must be selected before any label is observed, the analogue of this greedy criterion selects the example that maximally increases the label entropy of the selected set. This greedy criterion in the non-adaptive case is well-known to be near-optimal: the label entropy of the selected examples is at least $(1 - 1/e)$ of the optimal set. This follows from a property satisfied by the entropy function called *submodularity*. Selecting a set with large label entropy is desirable, as the chain rule of entropy implies that maximizing the label entropy of the selected set will minimize the conditional label entropy of the remaining examples. It would be desirable to have a similar near-optimal performance guarantee for the adaptive case where the label is provided after every example is selected. Whether the greedy maximum entropy criterion provides such a guarantee was not known (Cuong et al., 2013), although it was suspected that it does not. In this paper, we show that the greedy algorithm, indeed, does not provide a constant factor approximation in the adaptive case.

Another commonly used greedy criterion is the *least confidence criterion*: select the example whose most likely label has the smallest probability (Lewis and Gale, 1994; Culotta and McCallum, 2005). In this paper, we show that this criterion provides a near-optimal adaptive algorithm for maximizing the worst-case version space reduction, where the version space is the probability of labelings that are consistent with the observed labels. This will be derived as the consequence of a more general result which shows such near-optimal approximation holds for utility functions that satisfy *pointwise submodularity* and *minimal dependency*. Pointwise submodular functions were previously studied in (Guillory and Bilmes, 2010) for active learning, but with a different objective function which focuses on identifying the true function.

The *Gibbs error criterion* was proposed in (Cuong et al., 2013) as an alternative uncertainty measure suitable for ac-

Table 1: Theoretical Properties of Greedy Criteria for Adaptive Active Learning

Criterion	Objective	Near-optimality	Property
Maximum entropy	Policy entropy	No constant factor approximation (this paper)	
Least confidence	Worst-case version space reduction	(1-1/e) factor approximation (this paper)	Pointwise monotone submodular
Maximum Gibbs error	Policy Gibbs error (expected version space reduction)	(1-1/e) factor approximation (Cuong et al., 2013)	Adaptive monotone submodular

tive learning. The criterion selects the example with the largest Gibbs error for labeling. The Gibbs error is the expected error of the Gibbs classifier, which predicts the label by sampling from the current label distribution. Gibbs error is a special case of Tsallis entropy, introduced in statistical mechanics (Tsallis and Brigatti, 2004) as a generalization of the Shannon entropy (which is used in the maximum entropy criterion). In (Cuong et al., 2013), Gibbs error was used as a lower bound to the Shannon entropy and was maximized in order to minimize the posterior conditional entropy. It was shown in (Cuong et al., 2013) that using the Gibbs error criterion achieves at least $(1 - 1/e)$ of the optimal policy Gibbs error, a performance measure for this criterion, given k queries in the adaptive case. This relies on the property that the version space reduction function is *adaptive submodular* (Golovin and Krause, 2011).

The results for the three commonly used greedy criteria are shown in Table 1.

The Gibbs error criterion can be seen as a greedy algorithm for sequentially maximizing the Gibbs error over the dataset. The Gibbs error of the dataset is the expected error of a Gibbs classifier that predicts using an entire labeling sampled from the prior label distribution for the entire dataset. Here, a labeling is considered incorrect if any example is incorrectly labeled by the Gibbs classifier. Predicting an entirely correct labeling of all examples is often unrealistic in practice, particularly after only a few examples are labeled. This motivates us to generalize the Gibbs error to handle different loss functions between labelings, e.g. Hamming loss which measures the Hamming distance between two labelings. We call the greedy criterion that uses general loss functions the *average generalized Gibbs error* criterion.

The corresponding performance measure for the average generalized Gibbs error criterion is the generalized policy Gibbs error, which is the expected value of the generalized version space reduction function. The generalized version space reduction function is an extension of the version space reduction function with general loss functions. We investigate whether the generalized version space reduction

function is adaptive submodular, as this property would provide a constant factor approximation for the average generalized Gibbs error criterion. Unfortunately, we can show that the generalized version space reduction function is not necessarily adaptive submodular, although it is adaptive submodular for the special case of the version space reduction function. Despite that, we show in our experiments that the average generalized Gibbs error criterion can perform well in practice, even when we do not know whether the corresponding utility function is adaptive submodular.

As in the case for the least confidence criterion, we also consider a worst-case setting for the generalized Gibbs error. The worst case against a target labeling can be severe, so we consider a variant: the total generalized version space reduction function. This function targets the sum of the remaining losses over all the remaining labelings, rather than against a single worst-case labeling. We call the corresponding greedy criterion the *worst-case generalized Gibbs error* criterion. It selects the example with maximum worst-case total generalized version space reduction as the next query. As the total generalized version space reduction function is pointwise submodular and satisfies the minimal dependency property, the method is guaranteed to be near-optimal. Our experiments show that the worst-case generalized Gibbs error criterion performs well in practice. For binary problems, the maximum entropy, least confidence, and Gibbs error criteria are all equivalent, and the worst-case generalized Gibbs error criterion outperforms them for most problems in our experiments.

2 PRELIMINARIES

Let \mathcal{X} be a finite set of items (or examples), and let \mathcal{Y} be a finite set of labels (or states). A *labeling* of \mathcal{X} is a function from \mathcal{X} to \mathcal{Y} , and a *partial labeling* is a partial function from \mathcal{X} to \mathcal{Y} . Each labeling of \mathcal{X} can be considered as a hypothesis in the hypothesis space $\mathcal{H} = \mathcal{Y}^{\mathcal{X}}$. In the Bayesian setting, there is a prior probability $p_0[h]$ on \mathcal{H} , and an unknown true hypothesis h_{true} is initially drawn from $p_0[h]$. After observing a labeled set (i.e. a partial labeling) \mathcal{D} ,

we can obtain the posterior $p_{\mathcal{D}}[h] = p_0[h|\mathcal{D}]$ using Bayes' rule.

For any $S \subseteq \mathcal{X}$ and any distribution p on \mathcal{H} , we write $p[\mathbf{y}; S]$ to denote the probability that a randomly drawn hypothesis from p assigns labels in the sequence \mathbf{y} to items in the sequence S . That is, $p[\mathbf{y}; S] \stackrel{\text{def}}{=} \sum_{h \in \mathcal{H}} p[h] \mathbb{P}[h(S) = \mathbf{y}|h]$, where we use the notation $h(S)$ to denote the sequence $(h(x_1), \dots, h(x_i))$ whenever S is a sequentially constructed set (x_1, \dots, x_i) , or simply the set $\{h(x) : x \in S\}$ if the items in S are not ordered. In our setting, h is a deterministic hypothesis, so $\mathbb{P}[h(S) = \mathbf{y}|h] = \mathbf{1}(h(S) = \mathbf{y})$, where $\mathbf{1}(\cdot)$ is the indicator function. Note that $p[\cdot; S]$ is a probability distribution on the set of all label sequences \mathbf{y} of S . For $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, we also write $p[y; x]$ for $p[\{y\}; \{x\}]$.

In practice, we often consider probabilistic models (like the naive Bayes models) which are used to generate labels for examples, and a prior is imposed on these models instead of on the labelings. In this case, we can follow the construction in the supplementary material of (Cuong et al., 2013) to construct an equivalent prior on the labelings and work with this induced prior.

We consider pool-based active learning with a fixed budget: given a budget of k queries, we aim to adaptively select from the pool \mathcal{X} the best k examples with respect to some objective function.¹ A pool-based active learning algorithm corresponds to a policy for choosing training examples from \mathcal{X} . A *policy* is a mapping from a partial labeling to the next unlabeled example to query. When the active learning policy chooses an unlabeled example, its label according to h_{true} will be revealed.

A policy can be represented by a policy tree in which each node corresponds to an unlabeled example to query, and edges below a node correspond to its labels. In this paper, we use policy and policy tree interchangeably. A policy can be non-adaptive or adaptive. In a non-adaptive policy, the observed labels are not taken into account when the policy chooses the next example. An adaptive policy, on the other hand, can use the observed labels to determine the next example to query. We will focus on adaptive policies in this paper.

Let Π_k be the set of policy trees of height k . Note that $\Pi_{|\mathcal{X}|}$ contains full policy trees, while Π_k with $k < |\mathcal{X}|$ contains partial policy trees. Following the insight in (Cuong et al., 2013), for any (full or partial) policy π , we define a probability distribution $p_0^\pi[\cdot]$ over the paths from the root to a leaf of π . Intuitively, $p_0^\pi[\rho]$ is the probability that the policy π follows the path ρ during its execution. This probability distribution is induced by the randomness of h_{true} and is

¹ In our setting, the usual objective of determining the true hypothesis h_{true} is infeasible unless the support of p_0 is small. When $p_0[h] > 0$ for all h , we need to query the whole pool \mathcal{X} in order to determine h_{true} .

defined as $p_0^\pi[\rho] \stackrel{\text{def}}{=} p_0[y_\rho; x_\rho]$, where x_ρ (resp. y_ρ) is the sequence of examples (resp. labels) along path ρ . Some objective functions for pool-based active learning can be defined using this probability distribution.

3 SUBMODULARITY

Our objective in active learning can often be stated as maximizing some average or worst-case performance with respect to some utility function $f(S)$ in the non-adaptive case, or $f(S, h)$ in the adaptive case, where S is the set of chosen examples. When $f(S)$ is submodular or $f(S, h)$ is adaptive submodular, greedy algorithms are known to be near-optimal (Nemhauser et al., 1978; Golovin and Krause, 2011). We shall briefly summarize some results about greedy optimization of submodular functions and adaptive submodular functions, then prove a new result about the worst-case near-optimality of a greedy algorithm for maximizing a pointwise submodular function.²

3.1 NEAR-OPTIMALITY OF SUBMODULAR MAXIMIZATION

A set function $f : 2^{\mathcal{X}} \rightarrow \mathbb{R}$ is *submodular* if it satisfies the following diminishing return property: for all $A \subseteq B \subseteq \mathcal{X}$ and $x \in \mathcal{X} \setminus B$,

$$f(A \cup \{x\}) - f(A) \geq f(B \cup \{x\}) - f(B).$$

The function f is called *monotone* if $f(A) \leq f(B)$ for all $A \subseteq B$.

To select a set of size k that maximizes a monotone submodular function, one greedy strategy is to iteratively select the next example x^* that satisfies

$$x^* = \arg \max_x \{f(S \cup \{x\}) - f(S)\}, \quad (1)$$

where S is the previously selected examples. The following theorem by Nemhauser et al. (1978) states the near-optimality of this greedy algorithm when maximizing a monotone submodular function.

Theorem 1 (Nemhauser et al. 1978). *Let f be a monotone submodular function such that $f(\emptyset) = 0$, and let S_k be the set of examples selected up to iteration k using the greedy criterion in Equation (1). Then for all $k > 0$, we have $f(S_k) \geq (1 - 1/e) \max_{|S|=k} f(S)$.*

3.2 NEAR-OPTIMALITY OF ADAPTIVE SUBMODULAR MAXIMIZATION

Adaptive submodularity (Golovin and Krause, 2011) is an extension of submodularity to the adaptive setting. For a partial labeling \mathcal{D} and a full labeling h , we write $h \sim \mathcal{D}$ to

² Note that our result can also be applied to settings other than active learning.

denote that \mathcal{D} is consistent with h . That is, $\mathcal{D} \subseteq h$ when we view a labeling as a set of (x, y) pairs. For two partial labelings \mathcal{D} and \mathcal{D}' , we call \mathcal{D} a sub-labeling of \mathcal{D}' , if $\mathcal{D} \subseteq \mathcal{D}'$.

We consider a utility function $f : 2^{\mathcal{X}} \times \mathcal{Y}^{\mathcal{X}} \rightarrow \mathbb{R}_{\geq 0}$ which depends on the examples selected and the true labeling of \mathcal{X} . For a partial labeling \mathcal{D} and an example x , we define $\Delta(x|\mathcal{D}) \stackrel{\text{def}}{=} \mathbb{E}_h[f(\text{dom}(\mathcal{D}) \cup \{x\}, h) - f(\text{dom}(\mathcal{D}), h) \mid h \sim \mathcal{D}]$, where the expectation is with respect to $p_0[h \sim \mathcal{D}]$ and $\text{dom}(\mathcal{D})$ is the domain of \mathcal{D} .

From the definitions in (Golovin and Krause, 2011), f is *adaptive submodular* with respect to p_0 if for all \mathcal{D} and \mathcal{D}' such that $\mathcal{D} \subseteq \mathcal{D}'$, and for all $x \in \mathcal{X} \setminus \text{dom}(\mathcal{D}')$, we have $\Delta(x|\mathcal{D}) \geq \Delta(x|\mathcal{D}')$. Furthermore, f is *adaptive monotone* with respect to p_0 if for all \mathcal{D} with $p_0[h \sim \mathcal{D}] > 0$ and for all $x \in \mathcal{X}$, we have $\Delta(x|\mathcal{D}) \geq 0$.

Let π be a policy for selecting the examples and x_h^π be the set of examples selected by π under the true labeling h . We define the expected utility of π as $f_{\text{avg}}(\pi) \stackrel{\text{def}}{=} \mathbb{E}[f(x_h^\pi, h)]$, where the expectation is with respect to $p_0[h]$. To adaptively select a set of size k that maximizes f_{avg} , one greedy strategy is to iteratively select the next example x^* that satisfies

$$x^* = \arg \max_x \Delta(x|\mathcal{D}), \quad (2)$$

where \mathcal{D} is the partial labeling that has already been observed. The following theorem by Golovin and Krause (2011) states the near-optimality of this greedy policy when f is adaptive monotone submodular.

Theorem 2 (Golovin and Krause 2011). *Let f be an adaptive monotone submodular function with respect to p_0 , π be the adaptive policy selecting k examples using Equation (2), and π^* be the optimal policy with respect to f_{avg} that selects k examples. Then for all $k > 0$, we have $f_{\text{avg}}(\pi) > (1 - 1/e)f_{\text{avg}}(\pi^*)$.*

3.3 NEAR-OPTIMALITY OF POINTWISE SUBMODULAR MAXIMIZATION

Theorem 2 gives near-optimal average-case performance guarantee for greedily optimizing an adaptive monotone submodular function. We now give a new near-optimal worst-case performance guarantee for greedily optimizing a pointwise monotone submodular function. A utility function $f : 2^{\mathcal{X}} \times \mathcal{Y}^{\mathcal{X}} \rightarrow \mathbb{R}_{\geq 0}$ is said to be *pointwise submodular* if the set function $f_h(S) \stackrel{\text{def}}{=} f(S, h)$ is submodular for all h . Similarly, f is *pointwise monotone* if $f_h(S)$ is monotone for all h .

When f is pointwise monotone submodular, the average utility $f_{\text{avg}}(S) = \mathbb{E}_{h \sim p_0}[f(S, h)]$ is monotone submodular, and thus the non-adaptive greedy algorithm is a near-optimal non-adaptive policy for maximizing $f_{\text{avg}}(S)$

(Golovin and Krause, 2011). However, we are more interested in the adaptive policies in this section.

For any partial labeling \mathcal{D} , any $x \in \mathcal{X} \setminus \text{dom}(\mathcal{D})$, and any $y \in \mathcal{Y}$, we write $\mathcal{D} \cup \{(x, y)\}$ to denote the partial labeling \mathcal{D} with an additional mapping from x to y .

We assume that for any $S \subseteq \mathcal{X}$ and any labeling h , the value of $f(S, h)$ does not depend on the labels of examples in $\mathcal{X} \setminus S$. We call this the *minimal dependency* property. Let us extend the definition of f so that its second parameter can be a partial labeling. The minimal dependency property implies that for any partial labeling \mathcal{D} and any labeling $h \sim \mathcal{D}$, we have $f(\text{dom}(\mathcal{D}), h) = f(\text{dom}(\mathcal{D}), \mathcal{D})$. Without this minimal dependency property, the theorem in this section may not hold. We will see some examples of functions that satisfy or do not satisfy the minimal dependency property in Section 4 and 5.

For a partial labeling \mathcal{D} and an example x , define

$$\delta(x|\mathcal{D}) \stackrel{\text{def}}{=} \min_{y \in \mathcal{Y}} \{f(\text{dom}(\mathcal{D}) \cup \{x\}, \mathcal{D} \cup \{(x, y)\}) - f(\text{dom}(\mathcal{D}), \mathcal{D})\}.$$

We consider the adaptive greedy strategy that iteratively selects the next example x^* satisfying

$$x^* = \arg \max_x \delta(x|\mathcal{D}), \quad (3)$$

where \mathcal{D} is the partial labeling that has already been observed. For any policy π , let $f_{\text{worst}}(\pi) \stackrel{\text{def}}{=} \min_h f(x_h^\pi, h)$ be the worst-case objective function. The following theorem states the near-optimality of the above greedy policy with respect to f_{worst} when f is pointwise monotone submodular.³

Theorem 3. *Let f be a pointwise monotone submodular function such that $f(\emptyset, h) = 0$ for all h , and f satisfies the minimal dependency property. Let π be the adaptive policy selecting k examples using Equation (3), and π^* be the optimal policy with respect to f_{worst} that selects k examples. Then for all $k > 0$, we have $f_{\text{worst}}(\pi) > (1 - 1/e)f_{\text{worst}}(\pi^*)$.*

The main idea in proving this theorem is to show that at every step, the greedy policy can cover at least $(1/k)$ -fraction of the optimal remaining utility. This property can be proven by replacing the current greedy step with the optimal policy and considering the adversary's path for this optimal policy. See Appendix A for a proof of this theorem.

We note that in the worst-case setting, Golovin and Krause (2011) also considered the problem of minimizing the number of queries needed to achieve a target utility value. However, their results mainly rely on the condition that the

³ Note that in the definition of $f_{\text{worst}}(\pi)$, h has to range over the set $\mathcal{Y}^{\mathcal{X}}$ of all possible labelings. Otherwise, Theorem 3 does not necessarily hold.

utility function is adaptive submodular, not the pointwise submodular condition considered in this section. It is also worth noting that our new greedy criterion in Equation (3) is different from the greedy criterion considered by Golovin and Krause (2011), which is essentially Equation (2). Thus, our result does not follow from their result and is developed using a different argument.

4 PROPERTIES OF GREEDY ACTIVE LEARNING CRITERIA

We now briefly introduce three greedy criteria that have been used for active learning: maximum entropy, maximum Gibbs error, and least confidence. These criteria are equivalent in the binary-class case (i.e. they all choose the same examples to query), but they are different in the multi-class case. We will prove some new properties of the maximum entropy and the least confidence criteria.

4.1 MAXIMUM ENTROPY

The maximum entropy criterion chooses the next example whose posterior label distribution has the maximum Shannon entropy (Settles, 2010). Formally, this criterion chooses the next example x^* that satisfies

$$x^* = \arg \max_x \mathbb{E}_{y \sim p_{\mathcal{D}}[y; x]} [-\ln p_{\mathcal{D}}[y; x]], \quad (4)$$

where $p_{\mathcal{D}}$ is the posterior obtained after observing the partial labeling \mathcal{D} . From (Cuong et al., 2013), it is desirable to maximize the policy entropy

$$H_{\text{ent}}(\pi) \stackrel{\text{def}}{=} \mathbb{E}_{\rho \sim p_0^\pi} [-\ln p_0^\pi[\rho]],$$

where the expectation is over all the paths in the policy tree of π , as maximizing the policy entropy will minimize the expected label entropy given the observations. Criterion (4) can be viewed as a greedy algorithm for maximizing the policy entropy.

Due to the monotonicity and submodularity of Shannon entropy (Fujishige, 1978), we can construct a non-adaptive greedy policy that achieves near-optimality with respect to the objective function H_{ent} in the non-adaptive setting. In the adaptive setting, however, it was previously unknown whether the maximum entropy criterion is near-optimal with respect to H_{ent} (Cuong et al., 2013).

We now show that, in general, the maximum entropy criterion may not be near-optimal with respect to the objective function H_{ent} (Theorem 4).

Theorem 4. *Let π be the adaptive policy in Π_k selecting examples using Equation (4), and π^* be the optimal adaptive policy in Π_k with respect to H_{ent} . For any $0 < \alpha < 1$, there exists a problem where $H_{\text{ent}}(\pi)/H_{\text{ent}}(\pi^*) < \alpha$.*

The main idea in proving this theorem is to construct a set of independent distractor examples that have highest entropy but provide no information about the true hypothesis. The greedy criterion is tricked to choose only these distractor examples. On the other hand, there is an identifier example which gives the identity of the true hypothesis but has a lower entropy than the distractor examples. Once the label of the identifier example is revealed, there will be a number of high entropy examples to query, so that the policy entropy achieved is higher than that of the greedy algorithm. See the supplement for a proof of this theorem.

4.2 MAXIMUM GIBBS ERROR

The maximum Gibbs error criterion chooses the next example whose posterior label distribution has the maximum Gibbs error (Cuong et al., 2013). Formally, this criterion chooses the next example x^* that satisfies

$$x^* = \arg \max_x \mathbb{E}_{y \sim p_{\mathcal{D}}[y; x]} [1 - p_{\mathcal{D}}[y; x]]. \quad (5)$$

This criterion attempts to greedily maximize the policy Gibbs error

$$H_{\text{gibbs}}(\pi) \stackrel{\text{def}}{=} \mathbb{E}_{\rho \sim p_0^\pi} [1 - p_0^\pi[\rho]],$$

which is a lower-bound of the policy entropy $H_{\text{ent}}(\pi)$.

It has been shown by Cuong et al. (2013, sup.) that the policy Gibbs error H_{gibbs} corresponds to the expected version space reduction in \mathcal{H} . Furthermore, the maximum Gibbs error criterion in Equation (5) corresponds to the algorithm that greedily maximizes the expected version space reduction. For $S \subseteq \mathcal{X}$ and $h \in \mathcal{H}$, the version space reduction function is defined as $f(S, h) \stackrel{\text{def}}{=} 1 - p_0[h(S); S]$.

Since the version space reduction function is adaptive monotone submodular (Golovin and Krause, 2011), the maximum Gibbs error criterion is near-optimal with respect to the objective function H_{gibbs} in both the non-adaptive and adaptive settings. That is, the greedy policy using Equation (5) has the policy Gibbs error within a factor $(1 - 1/e)$ of the optimal policy (Cuong et al., 2013).

4.3 LEAST CONFIDENCE

The least confidence criterion chooses the next example whose most likely label has minimal posterior probability (Lewis and Gale, 1994; Culotta and McCallum, 2005). Formally, this criterion chooses the next examples x^* that satisfies

$$x^* = \arg \min_x \{ \max_{y \in \mathcal{Y}} p_{\mathcal{D}}[y; x] \}. \quad (6)$$

Note that $x^* = \arg \max_x \{ 1 - \max_y p_{\mathcal{D}}[y; x] \}$. Thus, the least confidence criterion greedily optimizes the error rate of the Bayes classifier on the distribution $p_{\mathcal{D}}[\cdot; x]$. In this section, we use the result in Section 3.3 to prove that

the least confidence criterion near-optimally maximizes the worst-case version space reduction.

For a policy π , we define the worst-case version space reduction objective as

$$H_{lc}(\pi) \stackrel{\text{def}}{=} \min_h f(x_h^\pi, h)$$

where f is the version space reduction function defined in Section 4.2. We note that f satisfies the minimal dependency property. It can also be shown that f is pointwise monotone submodular, and the least confidence criterion is equivalent to the criterion in Equation (3). Thus, it follows from Theorem 3 that the least confidence criterion is near-optimal with respect to the objective function H_{lc} (Theorem 5). See the supplement for a proof.

Theorem 5. *Let π be the adaptive policy in Π_k selecting examples using Equation (6), and π^* be the optimal adaptive policy in Π_k with respect to H_{lc} . For all $k > 0$, we have $H_{lc}(\pi) > (1 - 1/e)H_{lc}(\pi^*)$.*

5 ACTIVE LEARNING WITH GENERAL LOSS

In this section, let us focus on the maximum Gibbs error criterion in Section 4.2. The policy Gibbs error objective H_{gibbs} can be written as $H_{\text{gibbs}}(\pi) = \mathbb{E}_{h \sim p_0}[f(x_h^\pi, h)]$, where f is the version space reduction function (Cuong et al., 2013, sup.). Note that $f(x_h^\pi, h)$ is the expected 0-1 loss that a random labeling drawn from p_0 differs from h on x_h^π . Because of the nature of 0-1 loss, even if the random labeling only differs from h on one element of x_h^π , it is counted as an error.

To overcome this disadvantage, we formulate a new objective function that can handle an arbitrary general loss function $L : \mathcal{Y}^{\mathcal{X}} \times \mathcal{Y}^{\mathcal{X}} \rightarrow \mathbb{R}_{\geq 0}$ satisfying the following two properties: $L(h, h') = L(h', h)$ for any two labelings h and h' of \mathcal{X} , and if $h = h'$ then $L(h, h') = 0$. For $S \subseteq \mathcal{X}$ and $h \in \mathcal{H}$, we define the *generalized version space reduction function*

$$f_L(S, h) \stackrel{\text{def}}{=} \mathbb{E}_{h' \sim p_0}[L(h, h') \mathbf{1}(h(S) \neq h'(S))].$$

Note that $f_L(S, h) = \sum_{h': h(S) \neq h'(S)} p_0[h'] L(h, h')$, which can be written as

$$\sum_{h'} p_0[h'] L(h, h') - \sum_{h': h(S) = h'(S)} p_0[h'] L(h, h').$$

If L is the 0-1 loss, i.e. $L(h, h') = \mathbf{1}(h \neq h')$, we have $f_{0-1}(S, h) = \sum_{h': h(S) \neq h'(S)} p_0[h']$, which is equal to the version space reduction function $f(S, h)$.

Our new objective is to maximize the expected value of the generalized version space reduction

$$H_L^{\text{avg}}(\pi) \stackrel{\text{def}}{=} \mathbb{E}_{h \sim p_0}[f_L(x_h^\pi, h)].$$

When L is the 0-1 loss, this objective function is equal to the policy Gibbs error $H_{\text{gibbs}}(\pi)$. Thus, we call $H_L^{\text{avg}}(\pi)$ the *generalized policy Gibbs error*.

5.1 AVERAGE-CASE CRITERION

To maximize $H_L^{\text{avg}}(\pi)$, a natural algorithm is to greedily maximize f_L at each step. Let \mathcal{D} be the previously observed partial labeling, this greedy criterion chooses the next example x^* that satisfies

$$x^* = \arg \max_x \mathbb{E}_{h \sim p_{\mathcal{D}}}[f_L(\text{dom}(\mathcal{D}) \cup \{x\}, h) - f_L(\text{dom}(\mathcal{D}), h)] \quad (7)$$

We call this criterion the *average generalized Gibbs error criterion*.

From the result in Section 3.2, if f_L is adaptive monotone submodular, then using the average generalized Gibbs error criterion is near-optimal. Theorem 6 below states this result, which is a direct consequence of Theorem 2.

Theorem 6. *Let π_L^{avg} be the adaptive policy in Π_k selecting examples using Equation (7), and π^* be the optimal adaptive policy in Π_k with respect to H_L^{avg} . If f_L is adaptive monotone submodular with respect to the prior p_0 , then $H_L^{\text{avg}}(\pi_L^{\text{avg}}) > (1 - 1/e)H_L^{\text{avg}}(\pi^*)$.*

Note that if L is 0-1 loss, then f_L is adaptive monotone submodular with respect to any prior. Unfortunately, in general, f_L may not be adaptive submodular with respect to a prior p_0 (Theorem 7). See the supplement for a proof.

Theorem 7. *Let p_0 be a prior with $p_0[h] > 0$ for all h . There exists a loss function L such that f_L is not adaptive submodular with respect to p_0 .*

In the supplementary material, we also discuss a sufficient condition for f_L to be adaptive monotone submodular with respect to p_0 , and hence satisfy the precondition in Theorem 6. However, it remains open whether this sufficient condition is true for any interesting loss function other than 0-1 loss.

5.2 WORST-CASE CRITERION

We have shown in Theorem 7 that f_L may not be adaptive submodular, and thus we may not always have a theoretical guarantee for the average generalized Gibbs error criterion. In this section, we will reconsider our objective in the worst case instead of the average case.

In the worst case, we may want to maximize the objective function $H_L^{\text{worst}}(\pi) \stackrel{\text{def}}{=} \min_h f_L(x_h^\pi, h)$. However, using this objective function may be too conservative since the generalized version space reduction is computed only from the losses between the surviving labelings⁴ and the worst-

⁴ The surviving labelings in $f_L(S, h)$ are the labelings consistent with h on S .

case labeling. Instead, we propose a less conservative objective function based on the losses among all the surviving labelings. Formally, we define the following *total generalized version space reduction* function

$$t_L(S, h) \stackrel{\text{def}}{=} \sum_{h'} \sum_{h''} p_0[h'] L(h', h'') p_0[h''] - \sum_{h': h'(S)=h(S)} \sum_{h'': h''(S)=h(S)} p_0[h'] L(h', h'') p_0[h''].$$

Our new objective is to maximize the following function called the *worst-case total generalized policy Gibbs error*

$$T_L^{\text{worst}}(\pi) \stackrel{\text{def}}{=} \min_h t_L(x_h^\pi, h).$$

To maximize T_L^{worst} , we propose a greedy algorithm that maximizes the worst-case total generalized version space reduction at every step. Note that $t_L(S, h)$ satisfies the minimal dependency property, i.e. its value does not depend on the labels of $\mathcal{X} \setminus S$ in h . So, for a partial labeling \mathcal{D} , we have $t_L(\text{dom}(\mathcal{D}), h) = t_L(\text{dom}(\mathcal{D}), \mathcal{D})$ for any $h \sim \mathcal{D}$. Using this notation, the greedy criterion for choosing the next example x^* can be written as

$$x^* = \arg \max_x \{ \min_{y \in \mathcal{Y}} [t_L(\text{dom}(\mathcal{D}) \cup \{x\}, \mathcal{D} \cup \{(x, y)\}) - t_L(\text{dom}(\mathcal{D}), \mathcal{D})] \} \quad (8)$$

where \mathcal{D} is the previously observed partial labeling. We call this criterion the *worst-case generalized Gibbs error* criterion. It can be shown that t_L is pointwise monotone submodular and satisfies the minimal dependency property for any loss function L . Furthermore, the criterion in Equation (8) is equivalent to the criterion in Equation (3). Thus, it follows from Theorem 3 that this greedy criterion is near-optimal with respect to the objective function $T_L^{\text{worst}}(\pi)$ (Theorem 8). See the supplement for a proof.

Theorem 8. *Let π_L^{worst} be the adaptive policy in Π_k selecting examples using Equation (8), and π^* be the optimal adaptive policy in Π_k with respect to T_L^{worst} . We have $T_L^{\text{worst}}(\pi_L^{\text{worst}}) > (1 - 1/e) T_L^{\text{worst}}(\pi^*)$.*

It is worth noting that, like t_L , the function f_L is also pointwise submodular for any loss function L . The proof for the pointwise submodularity of f_L is essentially similar to the proofs that f and t_L are pointwise submodular in Theorem 5 and Theorem 8 (see the supplement for a proof of this claim). However, f_L does not satisfy the minimal dependency property. Besides, Theorem 7 also shows that f_L may not be adaptive submodular. Thus, this is an example that a pointwise submodular function is not necessarily adaptive submodular, and we may not be able to use Golovin and Krause (2011)’s result to obtain a result in the average case for pointwise submodular functions.

5.3 COMPUTING THE CRITERIA

In this section, we discuss the computations of the criteria in Equation (7) and Equation (8). First, we give two

propositions below regarding these equations. See the supplement for proofs.

Proposition 1. *The selected example x^* in Equation (7) is equal to*

$$\arg \min_x \sum_y \mathbb{E}_{h, h' \sim p_{\mathcal{D}}} [L(h, h') \mathbf{1}(h(x) = h'(x) = y)].$$

Proposition 2. *The selected example x^* in Equation (8) is equal to*

$$\arg \min_x \{ \max_y \mathbb{E}_{h, h' \sim p_{\mathcal{D}}} [L(h, h') \mathbf{1}(h(x) = h'(x) = y)] \}.$$

From these two propositions, we can compute Equation (7) and Equation (8) by estimating the expectation $\mathbb{E}_{h, h' \sim p_{\mathcal{D}}} [L(h, h') \mathbf{1}(h(x) = h'(x) = y)]$ for each $y \in \mathcal{Y}$. This estimation can be done by sampling from the posterior.

We can sample directly from $p_{\mathcal{D}}$ two sets H and H' which contain samples of h and h' respectively. Then, the expectation $\mathbb{E}_{h, h' \sim p_{\mathcal{D}}} [L(h, h') \mathbf{1}(h(x) = h'(x) = y)]$ can be approximated by

$$\frac{1}{|H| \times |H'|} \sum_{h \in H} \sum_{h' \in H'} L(h, h') \mathbf{1}(h(x) = h'(x) = y).$$

Note that this approximation only requires samples of the labelings from the posterior, and we do not need to explicitly maintain the set of all labelings which may be exponentially large. In the case when the labelings are generated by probabilistic models following some prior distribution, sampling from $p_{\mathcal{D}}$ may be difficult. A simple approximation is to sample H and H' from the MAP model.

6 EXPERIMENTS

Experimental results comparing the maximum entropy criterion, the maximum Gibbs error criterion, and the least confidence criterion were reported in (Cuong et al., 2013). In this section, we only focus on the active learning criteria with general loss functions, and conduct experiments with two common loss functions used in practice: the Hamming loss and the F_1 loss. For two labelings h and h' (viewing them as label vectors), the Hamming loss is the Hamming distance between them, and the F_1 loss is $1 - F_1(h, h')$ where $F_1(h, h') \in [0, 1]$ is the F_1 score between h and h' .

We experiment with various binary-class tasks from the UCI repository (Bache and Lichman, 2013) and the 20Newsgroups dataset (Joachims, 1996). We use the binary-class logistic regression as our model, and compare the active learners using the greedy criteria in Section 5.1 and 5.2 with the passive learner (Pass) and the maximum Gibbs error active learner (Gibbs). The maximum Gibbs error criterion is estimated from Equation (5) using the MAP

Table 2: AUC for Accuracy and F_1 on UCI Datasets

Dataset	Accuracy				F_1			
	Pass	Gibbs	WorstH	AvgH	Pass	Gibbs	WorstF	AvgF
Adult	74.81	73.94	77.81	77.72	82.00	81.12	85.15	84.57
Breast cancer	89.81	88.90	90.66	89.96	93.42	92.80	94.09	94.91
Diabetes	64.59	68.57	67.03	68.90	36.61	42.56	48.34	42.02
Ionosphere	78.31	82.96	84.77	83.79	63.99	72.57	72.19	72.93
Liver disorders	66.91	66.65	67.25	68.09	72.07	73.83	75.94	74.70
Mushroom	75.01	85.01	89.50	80.43	66.99	83.13	73.21	82.96
Sonar	65.75	68.76	67.58	66.37	71.84	75.31	73.92	73.48
Average	73.60	76.40	77.80	76.47	69.56	74.47	74.69	75.08

Table 3: AUC for Accuracy and F_1 on 20Newsgroups Dataset

Task	Accuracy				F_1			
	Pass	Gibbs	WorstH	AvgH	Pass	Gibbs	WorstF	AvgF
alt.atheism/comp.graphics	85.34	86.76	87.21	86.71	87.38	88.77	88.89	89.87
talk.politics.guns/talk.politics.mideast	73.37	80.75	75.03	77.03	77.46	82.23	79.72	79.88
comp.sys.mac.hardware/comp.windows.x	78.36	79.84	80.20	78.05	79.58	80.22	76.43	79.31
rec.motorcycles/rec.sport.baseball	82.34	82.44	85.37	83.27	80.74	83.06	84.48	83.97
sci.crypt/sci.electronics	72.75	77.07	77.83	78.71	67.53	73.92	73.82	77.69
sci.space/soc.religion.christian	80.96	85.58	87.35	87.84	79.95	84.51	86.05	87.16
soc.religion.christian/talk.politics.guns	82.10	84.01	85.81	85.83	80.43	79.24	83.37	82.46
Average	79.32	82.35	82.69	82.49	79.01	81.70	81.82	82.91

hypothesis. Note that the maximum Gibbs error criterion is equivalent to the maximum entropy and the least confidence criteria in this case since the tasks are binary-class.

We estimate the average-case criteria (AvgH and AvgF) in Section 5.1 and the worst-case criteria (WorstH and WorstF) in Section 5.2 using the approximation in Section 5.3 with the MAP hypothesis. AvgH and WorstH use the Hamming loss, while AvgF and WorstF use the F_1 loss. We compare the AUCs (area under the curve) for the accuracy scores of Pass, Gibbs, AvgH, and WorstH. We also compare the AUCs for the F_1 scores of Pass, Gibbs, AvgF, and WorstF.

The AUCs are computed from the first 150 examples and normalized so that their ranges are from 0 to 100. We randomly choose the first 10 examples as a seed set. We use the same seed set for all the algorithms.

The detailed procedure to compute the AUCs for our experiments is as follows. We sequentially choose 10 (seed size), 11, ..., 150 training examples using active learning or passive learning. Then for each training size, we train a model and compute its score (accuracy or F_1) on a separate test set. Using these scores, we can compute the AUCs. We

use the AUC scores because we want to compare the whole learning curves from choosing 10 to 150 training examples, not just the scores at any single point (e.g. 150 examples). This is consistent with previous works such as (Settles and Craven, 2008) and (Cuong et al., 2013).

The results for the UCI datasets are given in Table 2. From Table 2, all the active learning algorithms perform better than passive learning in terms of accuracy. On average, WorstH and AvgH perform slightly better than Gibbs, and WorstH achieves the best average AUC for accuracy. In addition, all the active learning algorithms also perform better than passive learning in terms of F_1 score. On average, WorstF and AvgF also perform slightly better than Gibbs, and AvgF achieves the best average AUC for F_1 score.

The results for the 20Newsgroups dataset are given in Table 3. From Table 3, all the active learning algorithms are better than passive learning in terms of accuracy. WorstH and AvgH are slightly better than Gibbs on average. Overall, WorstH achieves the best average AUC for accuracy. In addition, the active learning algorithms are also better than passive learning in terms of F_1 score. WorstF and AvgF are also slightly better than Gibbs, and AvgF has the best average AUC for F_1 score.

In both datasets, using the Hamming loss or F_1 loss is better than using the 0-1 loss (the Gibbs criterion). Furthermore, the worst-case criterion with Hamming loss achieves the best average scores in terms of accuracy, while the average-case criterion with F_1 loss achieves the best average scores in terms of F_1 .

7 CONCLUSION

We have discussed several theoretical properties of greedy algorithms for active learning. In particular, we proved a negative result for the maximum entropy criterion and a near-optimality result for the least confidence criterion in the worst case. We also considered active learning with general losses and proposed two greedy algorithms, one of which is for the average case and the other is for the worst case. Our experiments show that the new algorithms perform well in practice.

A APPENDIX: PROOF OF THEOREM 3

Let π and π^* be the policies as in the statement of Theorem 3. Let $h_\pi = \arg \min_h f(x_h^\pi, h)$. Then we have $f_{\text{worst}}(\pi) = f(x_{h_\pi}^\pi, h_\pi)$. Note that h_π corresponds to a path from the root to a leaf of the policy tree of π . Let the examples and labels along the path h_π (from the root of the tree to a leaf) be: $h_\pi \stackrel{\text{def}}{=} \{(x_1, y_1), (x_2, y_2), \dots, (x_k, y_k)\}$.

Since f satisfies the minimal dependency property, let us abuse the notation and write $f(\{x_t\}_{t=1}^i, \{y_t\}_{t=1}^i)$ to denote $f(\{x_t\}_{t=1}^i, h_\pi)$. Define

$$u_i \stackrel{\text{def}}{=} f(\{x_t\}_{t=1}^i, \{y_t\}_{t=1}^i) - f(\{x_t\}_{t=1}^{i-1}, \{y_t\}_{t=1}^{i-1})$$

$$v_i \stackrel{\text{def}}{=} \sum_{t=1}^i u_t \quad \text{and} \quad z_i \stackrel{\text{def}}{=} f_{\text{worst}}(\pi^*) - v_i.$$

We prove the following claims.

Claim 1. For all i , we have $u_{i+1} \geq z_i/k$.

Proof. Consider the case that after observing $(x_1, y_1), \dots, (x_i, y_i)$, we run the policy π^* from its root and only follow the paths consistent with $(x_1, y_1), \dots, (x_i, y_i)$ down to a leaf. In this case, all the paths of the policy π^* must obtain a value at least $z_i = f_{\text{worst}}(\pi^*) - v_i$, because running π^* without any observation would obtain at least $f_{\text{worst}}(\pi^*)$ and the observations $(x_1, y_1), \dots, (x_i, y_i)$ cover a value v_i .

Now we consider the adversary's path of the policy π^* in this scenario which is defined as

$$h^{\text{adv}} \stackrel{\text{def}}{=} \{(x_1^{\text{adv}}, y_1^{\text{adv}}), (x_2^{\text{adv}}, y_2^{\text{adv}}), \dots, (x_k^{\text{adv}}, y_k^{\text{adv}})\},$$

where $y_j^{\text{adv}} = \arg \min_y \{f(\{x_t\}_{t=1}^i \cup \{x_t^{\text{adv}}\}_{t=1}^{j-1} \cup \{x_j^{\text{adv}}\}, \{y_t\}_{t=1}^i \cup \{y_t^{\text{adv}}\}_{t=1}^{j-1} \cup \{y_j^{\text{adv}}\}) - f(\{x_t\}_{t=1}^i \cup \{x_t^{\text{adv}}\}_{t=1}^{j-1}, \{y_t\}_{t=1}^i \cup \{y_t^{\text{adv}}\}_{t=1}^{j-1})\}$

if x_j^{adv} has not appeared in $\{x_1, \dots, x_i\}$. Otherwise, if $x_j^{\text{adv}} = x_t$ for some $t \in \{1, \dots, i\}$, then $y_j^{\text{adv}} = y_t$. From the previous discussion, h^{adv} covers a value of at least z_i in k steps. Thus, one of its steps must cover a value of at least z_i/k .

Hence, what remains is to show that doing the greedy step in π after observing $(x_1, y_1), \dots, (x_i, y_i)$ is better than any single step along h^{adv} . In the trivial case where $(x_j^{\text{adv}}, y_j^{\text{adv}}) \in \{(x_1, y_1), \dots, (x_i, y_i)\}$, we obtain nothing in this step since $(x_j^{\text{adv}}, y_j^{\text{adv}})$ has already been observed. Thus, the above is true in this case. In the non-trivial case,

$$\begin{aligned} & u_{i+1} \\ &= f(\{x_t\}_{t=1}^{i+1}, \{y_t\}_{t=1}^{i+1}) - f(\{x_t\}_{t=1}^i, \{y_t\}_{t=1}^i) \\ &\geq \min_y \{f(\{x_t\}_{t=1}^i \cup \{x_{i+1}\}, \{y_t\}_{t=1}^i \cup \{y\}) \\ &\quad - f(\{x_t\}_{t=1}^i, \{y_t\}_{t=1}^i)\} \\ &\geq \min_y \{f(\{x_t\}_{t=1}^i \cup \{x_j^{\text{adv}}\}, \{y_t\}_{t=1}^i \cup \{y\}) \\ &\quad - f(\{x_t\}_{t=1}^i, \{y_t\}_{t=1}^i)\} \\ &\geq \min_y \{f(\{x_t\}_{t=1}^i \cup \{x_t^{\text{adv}}\}_{t=1}^{j-1} \cup \{x_j^{\text{adv}}\}, \\ &\quad \{y_t\}_{t=1}^i \cup \{y_t^{\text{adv}}\}_{t=1}^{j-1} \cup \{y_j^{\text{adv}}\}) \\ &\quad - f(\{x_t\}_{t=1}^i \cup \{x_t^{\text{adv}}\}_{t=1}^{j-1}, \{y_t\}_{t=1}^i \cup \{y_t^{\text{adv}}\}_{t=1}^{j-1})\} \\ &= f(\{x_t\}_{t=1}^i \cup \{x_t^{\text{adv}}\}_{t=1}^{j-1} \cup \{x_j^{\text{adv}}\}, \\ &\quad \{y_t\}_{t=1}^i \cup \{y_t^{\text{adv}}\}_{t=1}^{j-1} \cup \{y_j^{\text{adv}}\}) \\ &\quad - f(\{x_t\}_{t=1}^i \cup \{x_t^{\text{adv}}\}_{t=1}^{j-1}, \{y_t\}_{t=1}^i \cup \{y_t^{\text{adv}}\}_{t=1}^{j-1}). \end{aligned}$$

Note that the second inequality is due to the greedy criterion, and the third inequality is due to the submodularity of f on the adversary path. Therefore, this claim is true. \square

Claim 2. For all $i \geq 0$, we have $z_i \leq (1 - \frac{1}{k})^i f_{\text{worst}}(\pi^*)$.

Proof. We prove this claim by induction. For $i = 0$, this holds because $z_0 = f_{\text{worst}}(\pi^*)$ by definition. Assume that $z_i \leq (1 - \frac{1}{k})^i f_{\text{worst}}(\pi^*)$, then due to Claim 1,

$$\begin{aligned} z_{i+1} &= f_{\text{worst}}(\pi^*) - v_{i+1} = f_{\text{worst}}(\pi^*) - v_i - u_{i+1} \\ &= z_i - u_{i+1} \leq z_i - \frac{z_i}{k} = (1 - \frac{1}{k})z_i \\ &\leq (1 - \frac{1}{k})^{i+1} f_{\text{worst}}(\pi^*). \end{aligned}$$

Therefore, this claim is true. \square

To prove Theorem 3, we apply Claim 2 with $i = k$ and have $z_k \leq (1 - \frac{1}{k})^k f_{\text{worst}}(\pi^*) < \frac{1}{e} f_{\text{worst}}(\pi^*)$. Hence, $f_{\text{worst}}(\pi) = v_k = f_{\text{worst}}(\pi^*) - z_k > (1 - \frac{1}{e}) f_{\text{worst}}(\pi^*)$.

Acknowledgements

This work is supported by the US Air Force Research Laboratory under agreement number FA2386-12-1-4031.

References

- Kevin Bache and Moshe Lichman. UCI machine learning repository. *Irvine, CA: University of California, School of Information and Computer Science*, 2013.
- Aron Culotta and Andrew McCallum. Reducing labeling effort for structured prediction tasks. In *Proceedings of the 20th National Conference on Artificial Intelligence*, pages 746–751, 2005.
- Nguyen Viet Cuong, Wee Sun Lee, Nan Ye, Kian Ming A. Chai, and Hai Leong Chieu. Active learning for probabilistic hypotheses using the maximum Gibbs error criterion. In *Advances in Neural Information Processing Systems*, pages 1457–1465, 2013.
- R.S. Forsyth. PC/Beagle Users Guide. *BUPA Medical Research Ltd*, 1990.
- Satoru Fujishige. Polymatroidal dependence structure of a set of random variables. *Information and Control*, 39(1): 55–72, 1978.
- Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42(1):427–486, 2011.
- R. Paul Gorman and Terrence J. Sejnowski. Analysis of hidden units in a layered network trained to classify sonar targets. *Neural Networks*, 1(1):75–89, 1988.
- Andrew Guillory and Jeff Bilmes. Interactive submodular set cover. In *Proceedings of the International Conference on Machine Learning*, pages 415–422, 2010.
- Thorsten Joachims. A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. DTIC Document, 1996.
- Ron Kohavi. Scaling up the accuracy of Naive-Bayes classifiers: a decision-tree hybrid. In *Proceedings of The Second International Conference on Knowledge Discovery and Data Mining*, 1996.
- David D Lewis and William A Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12, 1994.
- Andrew McCallum and Kamal Nigam. Employing EM and pool-based active learning for text classification. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 350–358, 1998.
- George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions. *Mathematical Programming*, 14(1):265–294, 1978.
- Jeffrey Curtis Schlimmer. Concept acquisition through representational adjustment. *University of California, Irvine*, 1987.
- Burr Settles. Active learning literature survey. *Computer Sciences Technical Report 1648, University of Wisconsin–Madison*, 2010.
- Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1070–1079, 2008.
- V. G. Sigillito, S. P. Wing, L. V. Hutton, and K. B. Baker. Classification of radar returns from the ionosphere using neural networks. *Johns Hopkins APL Technical Digest*, pages 262–266, 1989.
- Jack W. Smith, J.E. Everhart, W.C. Dickson, W.C. Knowler, and R.S. Johannes. Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. In *Proceedings of the Annual Symposium on Computer Application in Medical Care*, pages 261–265, 1988.
- Constantino Tsallis and Edgardo Brigatti. Nonextensive statistical mechanics: A brief introduction. *Continuum Mechanics and Thermodynamics*, 16(3):223–235, 2004.
- William H. Wolberg and Olvi L. Mangasarian. Multisurface method of pattern separation for medical diagnosis applied to breast cytology. *Proceedings of the National Academy of Sciences*, 87(23):9193–9196, 1990.

A Permutation-Based Kernel Conditional Independence Test

Gary Doran^{*‡}

Krikamol Muandet[‡]

Kun Zhang[‡]

Bernhard Schölkopf[‡]

^{*}Case Western Reserve University
Cleveland, Ohio 44106, USA

gary.doran@case.edu

[‡]Max Planck Institute for Intelligent Systems
72076 Tübingen, Germany

{krikamol, kzhang, bs}@tuebingen.mpg.de

Abstract

Determining conditional independence (CI) relationships between random variables is a challenging but important task for problems such as Bayesian network learning and causal discovery. We propose a new kernel CI test that uses a single, learned permutation to convert the CI test problem into an easier two-sample test problem. The learned permutation leaves the joint distribution unchanged if and only if the null hypothesis of CI holds. Then, a kernel two-sample test, which has been studied extensively in prior work, can be applied to a permuted and an unpermuted sample to test for CI. We demonstrate that the test (1) easily allows the incorporation of prior knowledge during the permutation step, (2) has power competitive with state-of-the-art kernel CI tests, and (3) accurately estimates the null distribution of the test statistic, even as the dimensionality of the conditioning variable grows.

1 INTRODUCTION

A distribution P_{xyz} over variables X , Y , and Z satisfies a *conditional independence* relationship $X \perp\!\!\!\perp Y \mid Z$ (“ X is conditionally independent of Y given Z ”) when the joint distribution factorizes as $P_{xyz} = P_{x|z} P_{y|z} P_z$, assuming the existence of conditional density functions. There are several other equivalent characterizations of conditional independence (Dawid, 1979). Determining whether such conditional independence relationships hold between variables is important for problems such as Bayesian network learning, causal discovery, and counterfactual analysis. Using a conditional independence test as a subroutine, the PC algorithm (Spirtes, Glymour, and Scheines, 2000), for example, can be used to determine a set of causal graphs based on the conditional independence relationships between variables. Moreover, counterfactual analysis often requires assumptions of ignorability, which involve

conditional independences among counterfactual variables (Rosenbaum and Rubin, 1983).

Numerous approaches exist to measure conditional dependence or test for conditional independence. For example, under the assumption of Gaussian variables with linear dependence relationships, *partial correlation* can be used to test for conditional independence (Baba, Shibata, and Sibuya, 2004). Another characterization of conditional independence is that $P_{x|yz} = P_{x|z}$. Some tests use this characterization to determine conditional independence by measuring the distance between estimates of these conditional densities (Su and White, 2008). When the conditioning variable is discrete, $X \perp\!\!\!\perp Y \mid Z$ if and only if $X \perp\!\!\!\perp Y \mid Z = z_i$ for every possible value z_i that Z takes. Permutation-based tests have been successfully applied to conditional independence testing in this discrete-variable case (Tsamardinos and Borboudakis, 2010). Other tests use this characterization by *discretizing* continuous conditioning variables and testing for independence within each discrete “bin” of Z (Margaritis, 2005).

Generally, conditional independence testing is a challenging problem (Bergsma, 2004). The “curse of dimensionality” in terms of the dimensionality of the conditioning variable Z can make the problem even more difficult to solve. To see why, first consider the case when Z takes a finite number of values $\{z_1, \dots, z_k\}$; then $X \perp\!\!\!\perp Y \mid Z$ if and only if $X \perp\!\!\!\perp Y \mid Z = z_i$ for each value z_i . Given a sample of size n , even if the data points are evenly distributed across values of Z , we must show independence within every subset of the sample with identical Z values using only approximately n/k points within each subset. When Z is real-valued and P_z is continuous, the observed values of Z are almost surely unique. To extend the procedure to the continuous case, we must infer conditional independence using nonidentical but nearby values of Z , where “nearby” must be quantified with some distance metric. Finding nearby points becomes difficult (without additional assumptions) as the dimensionality of Z grows. To guarantee that conditional independence reduces to unconditional independence between X and Y within each subset,

we need a large number of subsets of Z . On the other hand, with many subsets, in each subset one may not have enough points to assess independence.

Recently, kernel-based tests have also been proposed for conditional as well as unconditional independence testing (see Section 3 for a more detailed discussion). Kernel functions can be used to implicitly map objects from an input space into a “feature space,” or reproducing kernel Hilbert space (RKHS) (Aizerman, Braverman, and Rozoner, 1964; Schölkopf and Smola, 2002). Some tests use the kernel mean embedding, which is an embedding of distributions into an RKHS (Berlinet and Thomas-Agnan, 2004; Smola et al., 2007; Sriperumbudur et al., 2010). When the kernel used is *characteristic*, the embeddings of two distributions are equal (under the distance metric imposed by the RKHS norm) if and only if the distributions are identical. For example, all *universal* kernels such as the radial basis function (RBF) kernel are characteristic (Sriperumbudur et al., 2010). The Hilbert–Schmidt independence criterion (HSIC) is an unconditional independence test that measures the distance in the RKHS between the embedding of a joint distribution and the embedding of the product of its marginal distributions. The HSIC can also be interpreted as the Hilbert–Schmidt norm of a cross-covariance operator, a generalization of the covariance matrix, between the RKHSs corresponding to the marginal distributions (Gretton et al., 2008). The intuition behind the test is that a joint distribution P_{xy} is equal to the product of its marginals if and only if $X \perp\!\!\!\perp Y$. The HSIC has been extended to the conditional independence setting using the norm of the conditional cross-covariance operator to measure conditional dependence (Fukumizu et al., 2008). However, this approach also degrades as the dimensionality of the conditioning variable increases. A more recent approach, the kernel conditional independence test (KCIT), proposed by Zhang et al. (2011), uses a characterization of conditional independence defined in terms of the partial association under all square-integrable functions relating the variables X , Y , and Z (Daudin, 1980). The test relaxes this characterization to use a smaller, but sufficiently rich class of functions from some universal RKHS. For this test, the distribution of the test statistic is known and can be estimated efficiently. However, as the dimensionality of the conditioning variable grows larger or the relationships between the variables grow more complex, the distribution of the KCIT test statistic under the null distribution becomes harder to accurately estimate in practice.

In contrast to a conditional independence test, a kernel two-sample test (Gretton et al., 2006, 2009, 2012a) merely tests whether two samples have been drawn from the same distribution. The two-sample problem is conceptually simpler than testing for conditional independence, and has been studied extensively in prior work. Thus, the behavior of the null distributions for two-sample test statistics are well-

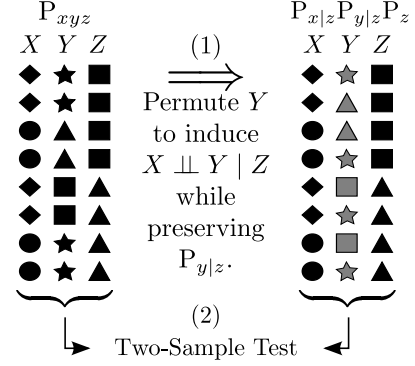


Figure 1: An overview of the proposed approach. First, we observe a sample from the joint distribution (left), and permute the sample to simulate a sample from the factorized distribution (right). The permutation is chosen to induce conditional independence while preserving $P_{x|z}$, $P_{y|z}$, and P_z . Then, a two-sample test is used to compare the permuted sample to an independent, unpermuted sample from the joint distribution.

understood.

We propose a new approach to test for conditional independence that uses a permutation to reduce the problem to a two-sample test. An overview of the approach is illustrated in Figure 1. First, a single, carefully chosen permutation is applied to a sample to simulate a sample from the factorized distribution $P_{x|z} P_{y|z} P_z$, which equals the underlying joint distribution if and only if the null hypothesis holds. Then, a kernel two-sample test (Gretton et al., 2012a) is performed between the permuted sample and an independent, unpermuted sample from the original distribution to determine whether the null hypothesis of conditional independence should be rejected. The approach permits various strategies for “learning” an appropriate permutation given prior knowledge about relationships between X , Y , and Z . The p -values for our test can be accurately, efficiently approximated using the approaches studied previously for kernel two-sample tests. We show using synthetic datasets that the proposed test has power competitive with state-of-the-art approaches, and can accurately estimate the distribution of the test statistic under the null hypothesis as the dimensionality of Z grows to produce a well-calibrated test. We also illustrate using a real-world dataset the practicability of the test for inferring conditional independence relationships.

2 DESCRIPTION OF THE TEST

In testing for *unconditional* independence, we observe an independent and identically distributed (i.i.d.) sample $\Omega = \{(x_i, y_i)\}_{i=1}^n$ drawn from P_{xy} . The variables X and Y are independent if and only if the joint distribution factorizes as

$P_{xy} = P_x P_y$. Here, P denotes a density function, but we use the same notation to represent the distribution itself. If we managed to draw a sample Ω' from $P_x P_y$, we could use a two-sample test between Ω and Ω' to determine whether to reject the null hypothesis $\mathcal{H}_0 : X \perp\!\!\!\perp Y$. Since we do not have access to the underlying joint distribution, but only a sample Ω , we must “simulate” a sample from the factorized distribution. By the i.i.d. assumption, the joint distribution of $(X_1, Y_1), \dots, (X_n, Y_n)$ is a product of identical factors $P_{xy}(X_i, Y_i)$. Hence for all i , X_i and Y_i have the same marginals P_x and P_y , respectively. Moreover, for $i \neq j$, we have $X_i \perp\!\!\!\perp Y_j$. If π is a permutation satisfying $\pi(i) \neq i$, we thus have $X_i \perp\!\!\!\perp Y_{\pi(i)}$, and the distribution of $(X_i, Y_{\pi(i)})$ must be $P_x(X_i) P_y(Y_{\pi(i)})$. Therefore, the permuted sample $(x_i, y_{\pi(i)})_{i=1}^n$ approximately simulates an i.i.d. sample from $P_x P_y$.¹

Below, we first discuss a way to extend the use of permutations to the conditional independence setting. Then, we show how to apply a kernel two-sample test to a permuted and an unpermuted sample to test for conditional independence. We describe how bootstrapping can be used to improve the power of the test. Given the two-sample test, we describe a kernel-based approach for learning an appropriate permutation.

2.1 PERMUTING FOR CONDITIONAL INDEPENDENCE

In this paper, for a joint distribution P_{xyz} over the variables X , Y , and Z , we are interested in determining whether $X \perp\!\!\!\perp Y \mid Z$, which occurs if and only if $P_{xyz} = P_{x|z} P_{y|z} P_z$. We observe an i.i.d. sample $\Omega = \{(x_i, y_i, z_i)\}_{i=1}^n$ drawn according to P_{xyz} . As above, if we were able to draw an independent sample Ω' from the factorized distribution $P_{x|z} P_{y|z} P_z$, we could use a two-sample test between Ω and Ω' to determine whether to reject the null hypothesis $\mathcal{H}_0 : X \perp\!\!\!\perp Y \mid Z$. Given the distributions $P_{x|z}$, $P_{y|z}$, and P_z , we could sample from the factorized distribution by first drawing $z_i \sim P_z$, and then $x_i \sim P_{x|z_i}$ and $y_i \sim P_{y|z_i}$. However, since we are given only a sample, we must “simulate” a sample from $P_{x|z} P_{y|z} P_z$. If the null hypothesis holds, one can consider each x_i and y_i in Ω as independently drawn from the conditional distributions $P_{x|z_i}$ and $P_{y|z_i}$, respectively. Suppose for some $i \neq j$, we find that $z_i = z_j$. In that case, we can proceed analogously to the unconditional case: we can swap the corresponding values y_i and y_j , breaking the dependence between X and Y , to obtain a joint observations

¹In the finite sample setting, this is only an approximation: while the permutation removes the dependence between X and its corresponding Y , it introduces a dependence to one of the other Y variables. In the limit $n \rightarrow \infty$, this becomes negligible (Janzing et al., 2013); moreover, in the limit we could waive the constraint $\pi(i) \neq i$ which we do not do in the present work since it is easy to implement.

(x_i, y_j, z_i) (and, likewise, (x_j, y_i, z_j)) drawn from the distribution $P_{x|z} P_{y|z} P_z$. Therefore, if we were able to (non-trivially) permute every y_i value so that the same permutation leaves the values of z_i in the sample invariant, then this would simulate i.i.d. draws from $P_{x|z} P_{y|z} P_z$. Unfortunately, when Z is continuous, the observed values of Z in Ω will be almost surely unique. In this case, we must “approximately” simulate a sample from $P_{x|z} P_{y|z} P_z$.

The procedure described above can be formalized as follows. Let the sample be expressed as $\Omega = (\mathbf{x}, \mathbf{y}, \mathbf{z})$, where \mathbf{x} , \mathbf{y} , and \mathbf{z} denote tuples of length n holding the sample elements for each of the variables (which might be multivariate), with ranges \mathcal{X} , \mathcal{Y} , and \mathcal{Z} . For the moment, we assume that \mathcal{X} , \mathcal{Y} , and \mathcal{Z} are equipped with addition and scalar multiplication. When we introduce the kernelization of the sample in Section 2.2, this assumption holds even when the sample elements are nonvectorial structured objects. Let \mathbf{P} be a linear transformation, represented as a matrix with nonnegative entries, that is defined to act on a sample as in: $\mathbf{P}\Omega \triangleq (\mathbf{x}, \mathbf{P}\mathbf{y}, \mathbf{z})$, where $\mathbf{P}\mathbf{y}$ is a tuple whose i^{th} element contains $\sum_j \mathbf{P}_{ij} y_j$. To preserve statistical properties of the sample, we cannot use a general linear transformation \mathbf{P} ; it must be a permutation matrix:

Proposition 1. *Let \mathcal{T} be the set of transformation such that for any $\mathbf{P} \in \mathcal{T}$ and sample \mathbf{y} of size n , $\text{mean}(\mathbf{P}\mathbf{y}) = \text{mean}(\mathbf{y})$ and $\|\text{var}(\mathbf{P}\mathbf{y})\|_{\text{HS}} = \|\text{var}(\mathbf{y})\|_{\text{HS}}$. Then \mathcal{T} is a set of permutation matrices of size n .²*

Essentially, the matrix \mathbf{P} must be stochastic to preserve the mean and orthogonal to preserve the variance, and these properties combined imply that it is a permutation. Given that \mathbf{P} is a permutation, we additionally require that $\text{Tr}(\mathbf{P}) = 0$, so that no element in the sample \mathbf{y} is permuted with itself (i.e., left unchanged). Otherwise, some dependence between x_i and y_i might remain. We use \mathcal{P} to denote the set of zero-trace permutations.

Ideally, we would further constrain \mathcal{P} so that all $\mathbf{P} \in \mathcal{P}$ satisfy $\mathbf{z} = \mathbf{P}\mathbf{z}$; that is, the values of \mathbf{z} are invariant under each permutation \mathbf{P} , or equivalently, we only permute the values of y_i that correspond to the same value of z_i . In the unconditional case, we can consider Z to be some constant variable, in which case any permutation is permitted. However, in the conditional case, this requirement is too restrictive so that the set of valid permutations is empty because each value of Z appears only once almost surely in the sample with continuous Z . Accordingly, we relax the problem to finding a permutation that enforces $\mathbf{z} \approx \mathbf{P}\mathbf{z}$. In particular, given some *distortion measure* $\delta : \mathcal{Z}^n \times \mathcal{Z}^n \rightarrow [0, \infty)$ that quantifies the discrepancy between permuted and unpermuted values of Z , we seek to optimize $\min_{\mathbf{P} \in \mathcal{P}} \delta(\mathbf{z}, \mathbf{P}\mathbf{z})$. For general classes of distortion measures, this optimiza-

²A proof can be found in the supplementary materials, available at http://engr.case.edu/doran_gary/publications.html.

tion problem is straightforward to solve. For example, let $d(z_i, z_j)$ be any symmetric pairwise distortion measure (e.g., a distance metric) and $\delta(\mathbf{z}, \mathbf{Pz}) = \sum_i d(z_i, (\mathbf{Pz})_i)$. Let \mathbf{D} be a matrix of pairwise distances between sample elements ($\mathbf{D}_{ij} = d(z_i, z_j)$). Since $\mathbf{P}_{ij} = 1$ if and only if z_i is permuted to z_j , $\delta(\mathbf{z}, \mathbf{Pz}) = \sum_{ij} \mathbf{P}_{ij} \mathbf{D}_{ij} = \text{Tr}(\mathbf{PD})$ and the distortion measure can be minimized using:

$$\min_{\mathbf{P} \in \mathcal{P}} \text{Tr}(\mathbf{PD}). \quad (1)$$

Relaxing \mathcal{P} to the set of doubly stochastic matrices (matrices whose rows and columns sum to one) with zero trace, the feasible region becomes the convex hull of permutation matrices, by the Birkhoff–von Neumann theorem (Birkhoff, 1946; von Neumann, 1953), subject to a linear constraint. Therefore, the simplex algorithm applied to Equation 1 returns a solution corresponding to a vertex of the feasible region, which is a permutation. The formulation in Equation 1 gives a general approach to permuting a sample, where the choice of distance metric d can encode some assumptions about the properties of the distributions $P_{x|z}$ or $P_{y|z}$. Below we discuss using \mathbf{P} to construct the test statistic and possible choices for d .

2.2 TEST STATISTIC AND NULL DISTRIBUTION

After learning an appropriate permutation, a two-sample test between a permuted and an unpermuted sample can be used to test the null hypothesis of conditional independence. A well-studied kernel-based two-sample test uses the maximum mean discrepancy (MMD) test statistic (Gretton et al., 2012a). The MMD employs *mean embeddings* of the two samples into some RKHS. Before describing the mean embedding, we introduce the notation used to “kernelize” the sample. Given the ranges \mathcal{X} , \mathcal{Y} , and \mathcal{Z} of the random variables X , Y , and Z , let $k_x(\cdot, \cdot)$, $k_y(\cdot, \cdot)$, and $k_z(\cdot, \cdot)$ be positive-definite *kernel* functions defined on these spaces ($k_x : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$, etc.). Corresponding to each kernel k_x is some *feature map* $\phi_x : \mathcal{X} \rightarrow \mathcal{H}_\mathcal{X}$ such that $k_x(x, x') = \langle \phi_x(x), \phi_x(x') \rangle$, where $\mathcal{H}_\mathcal{X}$ is the RKHS or *feature space* of k_x . We use a product of individual kernels to define the kernel on joint spaces; e.g., $k_{xyz}((x, y, z), (x', y', z')) = k_x(x, x')k_y(y, y')k_z(z, z')$ is a kernel over $\mathcal{X} \times \mathcal{Y} \times \mathcal{Z}$ with feature map $\phi_x \otimes \phi_y \otimes \phi_z$, where \otimes denotes the tensor product. Given that k_x , k_y , and k_z are translation-invariant characteristic kernels, the product kernel is also characteristic under mild assumptions (Sriperumbudur et al., 2010).

By mapping our sample into a feature space as $\{(\phi_x(x_i), \phi_y(y_i), \phi_z(z_i))\}_{i=1}^n$, we can treat each sample element as a vector (which is infinite-dimensional for the characteristic kernels used below), even when the underlying distribution over X , Y , and Z is over arbitrary sets of objects on which kernels are defined. In matrix notation, we can express the mapped sample as $(\Phi_x(\mathbf{x}), \Phi_y(\mathbf{y}), \Phi_z(\mathbf{z}))$, and the permuted sample in the

feature space as $\mathbf{P}\Omega = (\Phi_x(\mathbf{x}), \mathbf{P}\Phi_y(\mathbf{y}), \Phi_z(\mathbf{z}))$ with the i^{th} element of $\mathbf{P}\Phi_y(\mathbf{y})$ equal to $\sum_j \mathbf{P}_{ij} \phi_y(y_j)$. The conditions on \mathbf{P} in Proposition 1 are still necessary, since the linear kernel with feature map $\phi_y : \mathcal{Y} \rightarrow \mathcal{H}_\mathcal{Y}$ is a special case to which Proposition 1 applies. In prior work (Sriperumbudur et al., 2010), $\text{mean}(\Phi_y(\mathbf{y}))$ is called the *empirical kernel mean embedding*, and is expressed with the notation $\hat{\mu}(\mathbf{y}) = \frac{1}{n} \sum_{y \in \mathbf{y}} \phi_y(y)$. Given the constraints on \mathbf{P} , $\Phi_y(\mathbf{P}\mathbf{y}) = \mathbf{P}\Phi_y(\mathbf{y})$ for any Φ_y , and the mean embedding is invariant under \mathbf{P} : $\hat{\mu}(\mathbf{y}) = \hat{\mu}(\mathbf{P}\mathbf{y})$. The notation $\hat{\mu}(\Omega)$ will be used to denote the mean embedding of an entire sample using the product kernel defined on the joint space.

Given the kernelization of the sample, the test statistic is computed as follows. The original sample Ω of n elements is randomly split in half to form the samples $\Omega^{(1)}$ and $\Omega^{(2)}$, to ensure independence between the permuted and unpermuted samples (a condition required by the two-sample test). Using the formulation in Equation 1, we learn a permutation that induces conditional independence in the second subsample $\Omega^{(2)}$. Finally, we compute the (biased) MMD test statistic as follows:

$$\begin{aligned} \text{MMD}(\Omega^{(1)}, \mathbf{P}\Omega^{(2)}) &= \left\| \hat{\mu}(\Omega^{(1)}) - \hat{\mu}(\mathbf{P}\Omega^{(2)}) \right\|_{\mathcal{H}}^2 \\ &= \frac{4}{n^2} \mathbf{1}^\top (\mathbf{K}^{(1)} + \mathbf{K}^{(2)} - 2\mathbf{K}^{(12)}) \mathbf{1}. \end{aligned} \quad (2)$$

Here, $\mathbf{1}$ is a vector of ones of an appropriate size, the matrices $\mathbf{K}^{(1)}$ and $\mathbf{K}^{(2)}$ are pairwise kernel matrices within the permuted and unpermuted samples, respectively, and $\mathbf{K}^{(12)}$ is the “cross” kernel matrix between the unpermuted and permuted samples. Since we use product kernels, the matrices can be expressed in terms of a Hadamard product between the original kernel matrices for each variable:

$$\begin{aligned} \mathbf{K}^{(1)} &= \mathbf{K}_x^{(1)} \odot \mathbf{K}_y^{(1)} \odot \mathbf{K}_z^{(1)} \\ \mathbf{K}^{(2)} &= \mathbf{K}_x^{(2)} \odot (\mathbf{P}\mathbf{K}_y^{(2)}\mathbf{P}^\top) \odot \mathbf{K}_z^{(2)} \\ \mathbf{K}^{(12)} &= \mathbf{K}_x^{(12)} \odot (\mathbf{K}_y^{(12)}\mathbf{P}^\top) \odot \mathbf{K}_z^{(12)}, \end{aligned}$$

where $(\mathbf{K}_x)_{ij} = k_x(x_i, x_j)$, and likewise for \mathbf{K}_y and \mathbf{K}_z .

The behavior of the MMD test statistic has been extensively studied in prior work (Gretton et al., 2006, 2009, 2012a), and there are numerous approaches to estimating the null distribution and computing a p -value for the test statistic. For example, the null distribution can be estimated via a bootstrapping approach in which (1) $\Omega^{(1)}$ and $\mathbf{P}\Omega^{(2)}$ are randomly shuffled together and then split into two again, and then (2) the test statistic is recomputed between the shuffled samples (Gretton et al., 2009). Steps (1) and (2) are repeated b times to obtain an empirical estimate of the null distribution. The null distribution can also be approximated using a Gamma distribution. This estimate is computationally more efficient to obtain, but can also be less accurate in some scenarios (Gretton et al., 2009). As we are interested in the small-sample case, we choose to use the more robust bootstrap estimate at the expense of more computation.

Algorithm 1 KCIPT: Kernel Conditional Independence Permutation Test

Require: Sample $\Omega = (\mathbf{x}, \mathbf{y}, \mathbf{z})$, Distortion measure δ , Significant level α , Outer bootstrap iterations B , Inner bootstrap iterations b , Monte Carlo iterations M

- 1: **for** Outer Bootstrap $1 \leq i \leq B$ **do**
- 2: Split sample evenly into $\Omega^{(1)}, \Omega^{(2)}$
- 3: Find permutation matrix \mathbf{P} for $\Omega^{(2)}$ using δ to compute \mathbf{D} and solving Equation 1.
- 4: $\text{MMD}[i] \leftarrow \text{MMD}(\Omega^{(1)}, \mathbf{P}\Omega^{(2)})$
- 5: **for** Inner Bootstrap $1 \leq j \leq b$ **do**
- 6: Shuffle, re-split $\Omega^{(1)}, \mathbf{P}\Omega^{(2)}$ to Ω', Ω'' .
- 7: $\text{inner_null}[i, j] \leftarrow \text{MMD}(\Omega', \Omega'')$
- 8: **end for**
- 9: **end for**
- 10: $\text{statistic} \leftarrow \text{mean}_{1 \leq i \leq B}(\text{MMD}[i])$
- 11: **for** Monte Carlo Iteration $1 \leq k \leq M$ **do**
- 12: **for** Outer Bootstrap $1 \leq i \leq B$ **do**
- 13: $r \leftarrow \text{random_integer}(1, b)$
- 14: $\text{samples}[i] \leftarrow \text{inner_null}[i, r]$
- 15: **end for**
- 16: $\text{outer_null}[k] \leftarrow \text{mean}_{1 \leq i \leq B}(\text{samples}[i])$
- 17: **end for**
- 18: $p\text{-value} \leftarrow 1 - \text{percentile}(\text{statistic}, \text{outer_null})$
- 19: **if** $p\text{-value} \geq \alpha$ **then**
- 20: Fail to Reject $\mathcal{H}_0 (X \perp\!\!\!\perp Y \mid Z)$
- 21: **else**
- 22: Reject \mathcal{H}_0 , Conclude $X \not\perp\!\!\!\perp Y \mid Z$
- 23: **end if**

A characteristic kernel must be used to ensure that the MMD test statistic is consistent (convergent to zero if and only if the two samples are drawn from the same distribution). A kernel k_{xyz} is said to be *characteristic* if the corresponding mean map is injective (Sriperumbudur et al., 2010). Several popular kernels are characteristic, including the Gaussian RBF kernel $k(x, x') = \exp(-\|x - x'\|_2^2 / 2\sigma^2)$, with bandwidth parameter σ . Given that the RBF kernel is used for the test, there is still a question of how to select the bandwidth parameter. We set σ to be the median pairwise distance between sample elements, which prior work shows to be an effective heuristic (Gretton et al., 2012a). Other strategies existing for selecting σ to improve the power of the test statistic (Sriperumbudur et al., 2009; Gretton et al., 2012b).

2.3 BOOTSTRAPPING THE TEST STATISTIC

As defined above, the test statistic relies on splitting a sample randomly in half, which reduces the power of the two-sample test. However, if we randomly split the sample many times to compute many test statistics, we can bootstrap the MMD statistic itself to recover some of the power lost due to splitting. An overview of the test with bootstrapping is given in Algorithm 1.

Let $\{(\Omega_i^{(1)}, \Omega_i^{(2)})\}_{i=1}^B$ be a set of random splits of the dataset, where B denotes the number of random splits. The bootstrapped test statistic is the average of individual MMD test statistics for each split: $\text{MMD}_{\text{boot}}(\Omega) = \frac{1}{B} \sum_{i=1}^B \text{MMD}(\Omega_i^{(1)}, \mathbf{P}_i \Omega_i^{(2)})$, where \mathbf{P}_i is the permutation learned for the i^{th} split. The null distribution of MMD_{boot} can be estimated via a Monte Carlo simulation by repeatedly averaging together the draws from each individual test statistic's null distribution. Specifically, the null distribution N_i is first estimated for each test statistic $\text{MMD}_b(\Omega_i^{(1)}, \mathbf{P}_i \Omega_i^{(2)})$. Then, M points are drawn from each of the B null distributions: $s_{ij} \sim N_i$, for $1 \leq i \leq B$, $1 \leq j \leq M$. The points are averaged so that the resulting sample $\{\frac{1}{B} \sum_{i=1}^B s_{ij}\}_{j=1}^M$ is used to estimate the null distribution of $\text{MMD}_{\text{boot}}(\Omega)$.

Since we are combining many tests, any systematic error in estimating the null distribution will be compounded. Accordingly, we choose to use the robust bootstrapping approach described in Section 2.2 with a large number of draws b to estimate each null distribution N_i . Note that the statistic bootstrapping procedure (the “outer” bootstrap) is separate from the bootstrapping used to estimate the null hypothesis (the “inner” bootstrap). In the first case, a new permutation is learned for each split to compute the test statistic. In the second case, the learned permutation for the given split is left fixed, and the permuted and unpermuted subsamples are shuffled together randomly to simulate the null hypothesis. Since each N_i is an empirical estimate using a set of observed test statistics, we draw from N_i by sampling with replacement from the underlying set. If desired, the inner bootstrap shown in Algorithm 1 can be replaced with some other estimate of the null distribution of each MMD test statistic.

2.4 LEARNING THE PERMUTATION

Given the description of our test procedure, we now return to the issue of learning a permutation. Intuitively, since the test statistic uses an RKHS distance between samples, we would like our distortion measure to also utilize the RKHS distance. Therefore, we choose $d(z_i, z_j) = \|\phi_z(z_i) - \phi_z(z_j)\|$. In fact, we show that minimizing Equation 1 with respect to the RKHS distortion measure leads to a consistent test statistic when the distortion converges to zero. That is, we would like for the MMD between permuted and unpermuted samples to converge to zero if and only if the null hypothesis $\mathcal{H}_0 : X \perp\!\!\!\perp Y \mid Z$ holds.

Definition 1. A test statistic is asymptotically consistent if it converges in probability to zero if and only if the null hypothesis holds.

Theorem 1. Let $\mathbf{D}_{ij}^{\text{RKHS}} = \|\phi_z(z_i) - \phi_z(z_j)\|$ be a pairwise RKHS distance matrix between Z values in a sample. The proposed test statistic (Equation 2) is asymptot-

ically consistent if the quantity $\min_{\mathbf{P} \in \mathcal{P}} \frac{1}{n} \text{Tr}(\mathbf{P}\mathbf{D}^{\text{RKHS}})$ converges in probability to zero as $n \rightarrow \infty$.

Proof. Intuitively, minimizing $\frac{1}{n} \text{Tr}(\mathbf{P}\mathbf{D}^{\text{RKHS}})$ minimizes a majorant of the MMD between the permuted and unpermuted joint samples $(\mathbf{P}\mathbf{y}, \mathbf{z})$ and (\mathbf{y}, \mathbf{z}) . When this value converges to zero in probability, then so does the MMD, which implies that the permuted sample embedding converges to the embedding of the factorized joint distribution.³ \square

The optimal choice of the distance metric $d(z_i, z_j)$ should depend on how Z influences X and Y . Consider an extreme case where all dimensions of Z except Z_1 are irrelevant to (independent from) X and Y given Z_1 . We aim to find the nearby points along Z_1 , which are not necessarily neighbors when all dimensions are included. In other words, we should exclude all those irrelevant dimensions of Z when calculating the distances between z_i . An example is shown in Figure 2, where Y is some linear function of only the first component of Z , plus some Gaussian noise. Sample elements within the “level sets” of the hyperplane (indicated in the figure) are approximately exchangeable.

Generally speaking, given prior knowledge about structure in the relationships between variables, better measures of distance can be employed when learning the permutation. For example, a well-studied assumption in causal discovery is that X and Y are continuous functions of Z plus some independent Gaussian noise (Hoyer et al., 2008). When this is true, $P_{y|z} = \mathcal{N}(f(z), \Sigma)$, where f is some continuous function relating Z and Y , and Σ is a covariance matrix. In this case, $P_{y|z_i} \approx P_{y|z_j}$ if $f(z_i) \approx f(z_j)$, so it makes sense to use the distance metric $d(z_i, z_j) = \|f(z_i) - f(z_j)\|_2$ when learning the permutation. Although f is unknown, it can be learned from the data; e.g., by using Gaussian Process (GP) regression (Rasmussen and Williams, 2006). Of course, the consistency of the test statistic when heuristics are used depends upon whether the assumptions made by the heuristics are satisfied by the underlying joint distribution. In our experimental results, described below, we find that the function-based distance heuristic adds power to the test for synthetic datasets in which X and Y are in fact noisy functions of Z .

3 RELATED WORK

A previous approach, the conditional HSIC (CHSIC), uses the Hilbert–Schmidt norm of the conditional cross-covariance operator, which is a measure of conditional covariance of the images of X and Y under functions f and g from RKHSs corresponding to some kernels defined on X and Y . When the RKHSs correspond to characteristic kernels, the operator norm is zero if and only if $X \perp\!\!\!\perp Y \mid Z$

³See supplementary materials for the full proof.

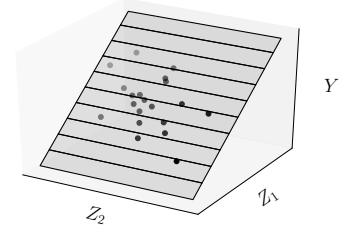


Figure 2: If Y is a function of Z plus noise, then many dimensions of Z might be irrelevant for determining conditional independence. In this example, Y is a noisy function of Z_1 , so sample elements within the level sets of the hyperplane are approximately exchangeable.

(Fukumizu et al., 2008). Since it is unknown how to analytically compute the null distribution of the CHSIC, the distribution is estimated using a bootstrapping approach. As described above for the MMD, a null distribution can be estimated by shuffling and recomputing the test statistic numerous times. In the conditional case, X and Y should only be shuffled when the corresponding Z values are near each other. Therefore, the values of Z are partitioned using a clustering algorithm, and bootstrap estimates are obtained by permuting Y values only within clusters (Fukumizu et al., 2008). Compared to our approach, the CHSIC has several disadvantages. The CHSIC requires *many* permutations to estimate the null distribution, whereas our approach only requires *one* carefully chosen permutation (per outer bootstrap iteration). Since the CHSIC clusters Z values to generate permutations, the permuted data points within each cluster have more widely varying values for Z , causing larger approximation errors. Finally, for high-dimensional datasets, finding an appropriate clustering algorithm becomes difficult, and the approximation quickly breaks down.

Other previous approaches to conditional independence testing use the partial association of regression functions relating X , Y , and Z (Huang, 2010; Zhang et al., 2011). In particular, the kernel-based KCIT (Zhang et al., 2011) is based on the following characterization of conditional independence: for any $f \in L^2_{XZ}$, and $g \in L^2_Y$, define $\tilde{f}(X, Z) = f(X, Z) - h_f(Z)$ and $\tilde{g}(Y, Z) = g(Y) - h_g(Z)$, where $h_f, h_g \in L^2_Z$ are regression functions of the values of f and g using only the variable Z . Then $X \perp\!\!\!\perp Y \mid Z$ if and only if for all $f \in L^2_{XZ}$, $g \in L^2_Y$, and \tilde{f}, \tilde{g} defined as above, $E[\tilde{f}\tilde{g}] = 0$ (Daudin, 1980). The KCIT relaxes the spaces of functions L^2_{XZ} , L^2_Y , and L^2_Z to be RKHSs corresponding to kernels defined on these variables. A universal kernel is required so that the RKHS for Z is dense in corresponding L^2_Z space. By contrast, the HSIC and our approach only require characteristic kernels, which need not be universal (Sriperumbudur et al., 2010).

4 EMPIRICAL EVALUATION

Our analysis suggests that by using a single permutation to compute the test statistic, our approach, the kernel conditional independence permutation test (KCIPT) will be more powerful than the CHSIC, which requires clustering the values of Z and many permutations in each cluster to estimate the null distribution. These permutations become difficult to find as the dimensionality of Z grows, as shown in prior work (Zhang et al., 2011). Furthermore, by using an MMD-based test statistic, the KCIPT can better estimate the null distribution than the KCIT in scenarios that require a careful choice of parameters. Finally, the outer bootstrapping procedure should improve the power of the KCIPT.

To empirically support our analysis, we implement KCIPT in MATLAB,⁴ and compare it to implementations of CHSIC and KCIT used in prior work (Zhang et al., 2011). We use two criteria for performance evaluation, type I error (the fraction of the time the null hypothesis H_0 is incorrectly rejected), and power (the fraction of the time H_0 is correctly rejected). Rather than choosing a specific significance level α at which to evaluate power and type I error, we record the p -values resulting from each test and analyze the behavior of the tests as α varies. For KCIPT, we use an RBF kernel $k(x, x') = \exp(-\|x - x'\|_2^2 / 2\sigma^2)$ for each variable, with bandwidth parameters σ_x , σ_y , and σ_z chosen using the “median” heuristic (Gretton et al., 2012a). For bootstrapping, we use parameters $B = 25$, $b = 10^4$, and $M = 10^4$. CHSIC and KCIT use the recommended parameters set in their implementations.

In order to characterize the power and type I error of the tests, we must evaluate the tests across many samples from the same underlying distribution. We use synthetic datasets from prior work for this purpose (Fukumizu et al., 2008; Zhang et al., 2011). Each dataset has a variant where the null hypothesis holds, for testing type I error, and where the null hypothesis does not hold, for testing power. We perform 300 tests for each condition, for each dataset described below.

Post-nonlinear Noise. The first dataset we use generates X and Y as functions of Z using a post-nonlinear noise model (Zhang and Hyvärinen, 2009; Zhang et al., 2011). In this generative process, the dimensionality of the conditioning variable Z grows, but only the first dimension Z_1 is relevant to the conditional independence of X and Y . Each of X and Y are determined using $G(F(Z_1) + E)$, where G and F are arbitrary smooth, nonlinear functions and E is a Gaussian noise variable. All dimensions of Z are i.i.d. Gaussian random variables. Since $X \perp\!\!\!\perp Y \mid Z$ by default, identical Gaussian noise is added to X and Y to produce a variant of the dataset for which $X \not\perp\!\!\!\perp Y \mid Z$. Because only

one conditioning variable is relevant to the problem, we expect that at least the KCIT and KCIPT with the function-distance distortion measure will be robust to increasing dimensionality, but that performance will degrade eventually.

Chaotic Times Series. The second dataset we use is a chaotic time series based on the Hénon map (Hénon, 1976). The two-dimensional variables $X = (X_t^{(1)}, X_t^{(2)})$ and $Y = (Y_t^{(1)}, Y_t^{(2)})$ are computed using only the values from the previous time step as follows:

$$\begin{aligned} X_t^{(1)} &= 1.4 - X_{t-1}^{(1)2} + 0.3X_{t-1}^{(2)} \\ Y_t^{(1)} &= 1.4 - \left[\gamma X_{t-1}^{(1)} Y_{t-1}^{(1)} + (1 - \gamma) Y_{t-1}^{(1)2} \right] + 0.3Y_{t-1}^{(2)} \\ X_t^{(2)} &= X_{t-1}^{(1)}, \quad Y_t^{(2)} = Y_{t-1}^{(1)}. \end{aligned}$$

The parameter γ controls the effect that previous values of X have on Y . To increase the difficulty of this task, two additional independent Gaussian noise variables with zero mean and standard deviation $\sigma = 0.5$ are concatenated to both X and Y . Here, conditional dependence and independence characterizes the causal influence from X to Y when $\gamma > 0$. Namely, in this dataset, $X_{t+1} \perp\!\!\!\perp Y_t \mid X_t$, but $Y_{t+1} \not\perp\!\!\!\perp X_t \mid Y_t$.

For each underlying joint distribution, we generate the cumulative density function (CDF) of the p -values obtained for each test across the 300 random samples. While the CDFs of the tests’ p -values are useful for understanding the global behavior of the tests,⁵ it is more succinct to summarize each curve with a single statistic. In prior work, the powers and type I errors at a particular, fixed value of α are used to summarize results (Fukumizu et al., 2008; Gretton et al., 2012a; Zhang et al., 2011). However, presenting results in this way can be misleading if one of the tests has an advantage at a particular value of α . Therefore, we use two statistics to summarize the power and type I error across values of α . When the test has high power, it correctly rejects the null hypothesis even when α is small. Therefore, the area under the CDF, or *power curve* is close to 1.0. On the other hand, when the null hypothesis is true, a *well-calibrated* test will produce uniformly-distributed p -values so that the type I error rate is equal to α . In this case, the CDF is a diagonal line with slope 1. To measure calibratedness, the Kolmogorov test can be used to quantify the difference between the empirically observed CDF and that for the uniform distribution. Since sample sizes are finite and null distributions are only approximately estimated, the null hypothesis of perfect calibratedness will likely be rejected by the Kolmogorov test after enough tests are performed. However, the relative (log) p -values corresponding to the Kolmogorov test can be used to compare calibratedness; larger p -values roughly correspond to better calibration.

⁴The code is available online at http://enr.case.edu/doran_gary/code.html

⁵See the supplementary materials for details.

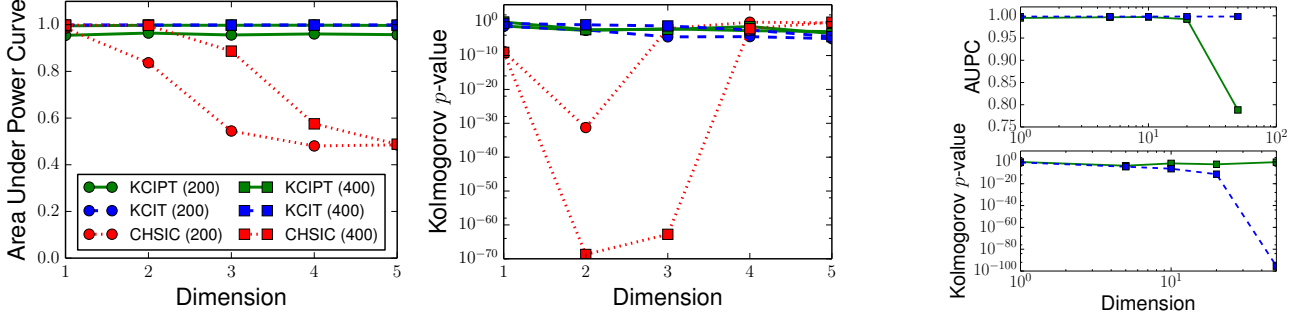


Figure 3: **(Left)** Summarized results for the post-nonlinear noise dataset. These results clearly show how the power of HSIC decreases as noise is added to the conditioning variable. **(Right)** A comparison of KCIT and KCIPT for high-dimensional datasets. The performance of the tests begin to degrade in different ways, with the power of KCIPT falling to chance levels while the KCIT becomes poorly calibrated between $D = 10$ and $D = 50$.

Figure 3 (left) shows results for the post-nonlinear noise dataset as the dimensionality D of the conditioning variable increases. Since Y is a function of Z , the function-distance distortion measure is used, as described in Section 2.4.⁶ Gaussian process regression is used to find the function f relating Z and Y . As observed in prior work (Zhang et al., 2011), the CHSIC approach is sensitive to the dimensionality of the conditioning variable, so power quickly decreases as D increases. With a dataset of size 200, KCIT is slightly more powerful than KCIPT, but the performance converges as the sample size increases to 400. Furthermore, the performance of KCIPT is preserved as dimensionality increases, since the regression-based distance effectively serves as dimensionality reduction on the conditioning variable.

Figure 3 (right) shows what happens to both KCIPT and KCIT as the dimensionality of the dataset continues to increase to $D = 50$; both tests fail by this point, but in different ways. KCIT becomes very poorly calibrated between $D = 10$ and $D = 50$, while the power of KCIPT degrades around the same dimensionality. We conjecture that the observed behavior is due to the differences in kernel parameter selection for each approach. The kernel values for KCIT are chosen heuristically depending on dataset size, but the test is only evaluated on low-dimensional datasets (Zhang et al., 2011). As dimensionality increases, the heuristic is less effective, and the test poorly estimates the null distribution. By contrast, the KCIPT uses the median heuristic, which automatically adjusts the kernel parameter as dataset size and dimensionality increase. Thus, the null distribution is correctly estimated, but the test statistic becomes less powerful on this dataset.

The results for the chaotic time series are shown in Figure 4. For this test, the RKHS distance is used as the distortion measure to learn the permutation. The behavior of the

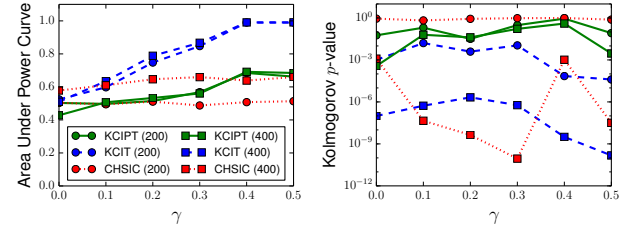


Figure 4: Results for the chaotic time series. As expected, the power of these tests increases as the conditional dependence controlled by γ increases. The KCIT is not well-calibrated on this dataset, and HSIC becomes less well-calibrated as sample size increases.

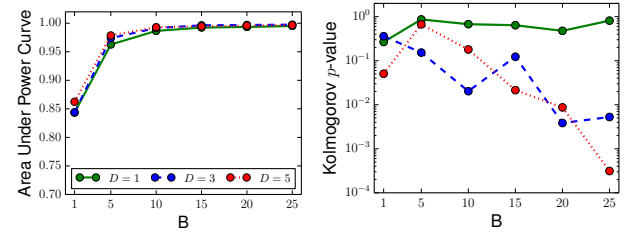


Figure 5: Effects of bootstrapping the test statistic with B iterations on the post-nonlinear noise dataset with $n = 400$. Bootstrapping increases the power of the test, but also decreases the calibration when dimensionality D of the conditioning variable increases. The observed effect is likely due to the approximation errors induced by the permutation leading to an over-rejection of the null hypothesis.

tests is shown as γ increases. In this noisy chaotic dataset, conditional dependence is more difficult to detect, and none of the techniques perform very well when γ is small. Although KCIT has the best performance in terms of power, it is poorly-calibrated as the sample size increases. In fact, both the CHSIC and KCIT become *less* well-calibrated as

⁶We compare this distortion measure with other choices in the supplementary materials.

sample size increases, suggesting systematic errors in null distribution estimation. For CHSIC, it appears that there are difficulties in finding permutations to estimate the null distribution, and for KCIT, the chaotic nature of the dataset might violate its assumption that variables are related by continuous, well-behaved functions.

Using the post-nonlinear noise dataset with $n = 400$, we also quantify the extent to which the outer bootstrapping procedure described in Section 2.3 improves the power of the test. Figure 5 shows the power and calibration of the test as the number of bootstraps B increases; $B = 1$ corresponds to no bootstrapping of the test statistic, and $B = 25$ is used in the previous experiments. Bootstrapping the test statistic does in fact increase power for this dataset. However, when the dimensionality of Z grows, the calibration of the test decreases. We believe that this behavior is a result of the approximation error induced by the permutation; as dimensionality increases, it becomes harder to find an appropriate permutation with a fixed sample size. However, we observe in Figure 3 (left) that the other tests also tend to be poorly calibrated on this dataset as the dimensionality of Z increases. These results do not suggest a general procedure for selecting B , but they illustrate that at least for the post-nonlinear noise data, there is a power-calibration trade-off involved in the use of bootstrapping.

Medical Data. Finally, we explore the application of the KCIPT to a real-world dataset used in prior work (Fukumizu et al., 2008). The data consists of three variables, creatinine clearance (C), digoxin clearance (D), and urine flow (U), measured on 35 patients. The ground truth, that $D \perp\!\!\!\perp U \mid C$, is known for this dataset. We try to recover this relationship using the PC algorithm (Spirtes, Glymour, and Scheines, 2000), with the KCIPT and $\alpha = 0.05$ used as a test for conditional independence. We choose $B = 10$, since it appears to be an effective setting that reduces the overall computation time (Figure 5). The output of the PC algorithm is the Markov equivalence class $D-C-U$, which contains the only causal structures (either $D \leftarrow C \leftarrow U$, $D \rightarrow C \rightarrow U$, or $D \leftarrow C \rightarrow U$) consistent with the ground truth conditional independence relationship and pairwise dependence relationships, assuming there are no unobserved confounding variables.

5 DISCUSSION

In relation to existing kernel-based conditional independence tests, a major advantage of KCIPT observed in our empirical analysis is its ability to accurately estimate the null distribution. Hence, we observe that KCIPT is well-calibrated across the synthetic datasets we study, even under the more extreme scenarios when the dimensionality of the conditioning variable is large or there are complex, nonlinear relationships between variables in the joint distribution. Our results align with those observed in

prior work, in which permutation-based conditional independence tests for datasets with discrete values were found to be well-calibrated with respect to asymptotic tests (Tsamardinos and Borboudakis, 2010). Additionally, using a well-calibrated conditional independence test produces more robust solutions in Bayesian network learning.

The need for conditional independence testing is ubiquitous in the sciences. Unfortunately, performing the test in practice is known to be very challenging. This work not only simplifies the problem, but also presents a general framework for conditional independence testing which can be extended immediately to numerous settings. Thus, there remain many interesting extensions and questions to study in future work, such as applications to non-i.i.d. data, different approaches for learning a permutation, and deciding which variable to permute in the asymmetric test statistic. Furthermore, we look forward to applying KCIPT to real-world datasets with more complex conditional dependence relationships.

6 CONCLUSION

In this work, we propose a new conditional independence test that employs a permutation to generate an artificial sample from a joint distribution for which the null hypothesis of the test holds. Effectively, we transform the conditional independence test into a two-sample test problem, which is easier to solve, well-studied, and scales to high-dimensional datasets. We use a kernel-based two sample test between an original sample and a permuted sample, which share the same distribution if and only if the conditional independence relationship holds. Prior knowledge about the joint distribution can be incorporated into the process of finding an appropriate permutation. The resulting test has power competitive with existing kernel-based approaches for conditional independence testing and better estimates the null distribution on the datasets used for evaluation. In future work, we will explore theoretical relationships between our approach and those using partial association and further investigate the use of our test for applications in causal discovery.

Acknowledgments

We thank the anonymous reviewers for their comments and suggestions.

References

- Aizerman, A.; Braverman, E. M.; and Rozoner, L. 1964. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and remote control* 25:821–837.
- Baba, K.; Shibata, R.; and Sibuya, M. 2004. Partial correlation and conditional correlation as measures of condi-

- tional independence. *Australian & New Zealand Journal of Statistics* 46(4):657–664.
- Bergsma, W. 2004. Testing conditional independence for continuous random variables. EURANDOM-report 2004-049.
- Berlinet, A., and Thomas-Agnan, C. 2004. *Reproducing kernel Hilbert spaces in probability and statistics*, volume 3. Springer.
- Birkhoff, G. 1946. Three observations on linear algebra. *Univ. Nac. Tucumán. Revista A* 5:147–151.
- Daudin, J. 1980. Partial association measures and an application to qualitative regression. *Biometrika* 67(3):581–590.
- Dawid, A. P. 1979. Conditional independence in statistical theory. *Journal of the Royal Statistical Society. Series B (Methodological)* 1–31.
- Fukumizu, K.; Gretton, A.; Sun, X.; and Schölkopf, B. 2008. Kernel measures of conditional dependence. In *Advances in Neural Information Processing Systems*, 489–496.
- Gretton, A.; Borgwardt, K. M.; Rasch, M.; Schölkopf, B.; and Smola, A. J. 2006. A kernel method for the two-sample-problem. In *Advances in Neural Information Processing Systems*, 513–520.
- Gretton, A.; Fukumizu, K.; Teo, C. H.; Song, L.; Schölkopf, B.; and Smola, A. J. 2008. A kernel statistical test of independence. In *Advances in Neural Information Processing Systems*, 585–592.
- Gretton, A.; Fukumizu, K.; Sriperumbudur, B. K.; et al. 2009. A fast, consistent kernel two-sample test. In *Advances in Neural Information Processing Systems*, 673–681.
- Gretton, A.; Borgwardt, K. M.; Rasch, M. J.; Schölkopf, B.; and Smola, A. 2012a. A kernel two-sample test. *The Journal of Machine Learning Research* 13:723–773.
- Gretton, A.; Sejdinovic, D.; Strathmann, H.; Balakrishnan, S.; Pontil, M.; Fukumizu, K.; and Sriperumbudur, B. K. 2012b. Optimal kernel choice for large-scale two-sample tests. In *Advances in Neural Information Processing Systems*, 1214–1222.
- Hénon, M. 1976. A two-dimensional mapping with a strange attractor. *Communications in Mathematical Physics* 50(1):69–77.
- Hoyer, P.; Janzing, D.; Mooij, J.; Peters, J.; and Schölkopf, B. 2008. Nonlinear causal discovery with additive noise models. In *Advances in Neural Information Processing Systems*, 689–696.
- Huang, T.-M. 2010. Testing conditional independence using maximal nonlinear conditional correlation. *The Annals of Statistics* 38(4):2047–2091.
- Janzing, D.; Balduzzi, D.; Grosse-Wentrup, M.; and Schölkopf, B. 2013. Supplement to: Quantifying causal influences. *The Annals of Statistics*. DOI: 10.1214/13-AOS1145SUPP.
- Margaritis, D. 2005. Distribution-free learning of Bayesian network structure in continuous domains. In *Proceedings of the National Conference on Artificial Intelligence*, volume 20, 825.
- Rasmussen, C., and Williams, C. 2006. *Gaussian Processes for Machine Learning*. Cambridge, Massachusetts, USA: MIT Press.
- Rosenbaum, P. R., and Rubin, D. B. 1983. The central role of the propensity score in observational studies for causal effects. *Biometrika* 70:41–55.
- Schölkopf, B., and Smola, A. 2002. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT Press.
- Smola, A.; Gretton, A.; Song, L.; and Schölkopf, B. 2007. A Hilbert space embedding for distributions. In *Algorithmic Learning Theory*, 13–31. Springer.
- Spirtes, P.; Glymour, C.; and Scheines, R. 2000. *Causation, prediction, and search*, volume 81. The MIT Press.
- Sriperumbudur, B. K.; Fukumizu, K.; Gretton, A.; Lanckriet, G. R.; and Schölkopf, B. 2009. Kernel choice and classifiability for RKHS embeddings of probability distributions. In *Advances in Neural Information Processing Systems*, 1750–1758.
- Sriperumbudur, B. K.; Gretton, A.; Fukumizu, K.; Schölkopf, B.; and Lanckriet, G. R. 2010. Hilbert space embeddings and metrics on probability measures. *The Journal of Machine Learning Research* 99:1517–1561.
- Su, L., and White, H. 2008. A nonparametric Hellinger metric test for conditional independence. *Econometric Theory* 24(4):829.
- Tsamardinos, I., and Borboudakis, G. 2010. Permutation testing improves Bayesian network learning. In *Machine Learning and Knowledge Discovery in Databases*. Springer. 322–337.
- von Neumann, J. 1953. A certain zero-sum two-person game equivalent to the optimal assignment problem. *Contributions to the Theory of Games* 2:5–12.
- Zhang, K., and Hyvärinen, A. 2009. On the identifiability of the post-nonlinear causal model. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*.
- Zhang, K.; Peters, J.; Janzing, D.; and Schölkopf, B. 2011. Kernel-based conditional independence test and application in causal discovery. In *Proceedings of the Twenty-Seventh Annual Conference on Uncertainty in Artificial Intelligence*, 804–813.

Parallel Markov Chain Monte Carlo for Pitman-Yor Mixture Models

Avinava Dubey

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Sinead A. Williamson

McCombs School of Business
University of Texas at Austin
Austin, TX 78712

Eric P. Xing

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

The Pitman-Yor process provides an elegant way to cluster data that exhibit power law behavior, where the number of clusters is unknown or unbounded. Unfortunately, inference in Pitman-Yor process-based models is typically slow and does not scale well with dataset size. In this paper we present new auxiliary-variable representations for the Pitman-Yor process and a special case of the hierarchical Pitman-Yor process that allows us to develop parallel inference algorithms that distribute inference both on the data space and the model space. We show that our method scales well with increasing data while avoiding any degradation in estimate quality.

1 INTRODUCTION

Bayesian nonparametric priors such as the Dirichlet process allow us to create flexible probabilistic models with an unbounded number of parameters. These models are appropriate when the latent dimensionality of our data is unknown or may grow with sample size. Unfortunately, inference in such models is often unwieldy, due to the high number of instantiated parameters and the need to discover the appropriate number of parameters for a given data set.

There has been growing interest in scalable inference algorithms for Bayesian nonparametric models. Earlier attempts at distributing inference were either highly model-specific (Doshi-Velez et al., 2009), or introduced additional approximation into the inference procedure and were mostly concentrated at distributed learning on data and not on the model (Asuncion et al., 2008). More recent approaches (Williamson et al., 2013; Lovell et al., 2012; Chang and Fisher III, 2013) have used both model- and data-parallel design to infer the latent structure without introducing additional approximation.

Most previous research on scalable inference in Bayesian

nonparametrics has focused on parallel inference for Dirichlet process-based models. Dirichlet process models are not ideal for modeling language, as they do not capture the power-law behavior often found in text data (Zipf, 1935). The Pitman-Yor process (Pitman et al., 1992; Pitman and Yor, 1997) is a two-parameter extension to the Dirichlet process that allows heavier-tailed distributions over partitions. It is therefore often used in text and language applications, because it more accurately matches the statistics of natural language (Goldwater et al., 2006; Teh, 2006a), and can be used to build hierarchical models for text that out-perform their Dirichlet process-based counterparts (Teh, 2006b; Wood et al., 2009; Blunsom and Cohn, 2011). However, inference remains a bottleneck.

In this paper, we address this issue using an approach pioneered for the Dirichlet process by Williamson et al. (2013) and Lovell et al. (2012): We construct an alternative representation of a nonparametric process that incorporates conditional independencies, and use these conditional independencies to divide our model (and in doing so, our data) into sub-models that can be learned in parallel.

The key to achieving this lies in the introduction of new representations for the Pitman-Yor process and a hierarchical extension, presented in Section 3. These representations afford the conditional independence structure required to develop model- and data-parallel inference algorithms, as demonstrated in Section 4. In Section 5, we perform a thorough evaluation of our inference algorithms and of the modeling assumptions made. We show that our hierarchical model, which is a special case of the hierarchical Pitman-Yor process (Teh, 2006b), is a good fit for natural language. We empirically demonstrate that we can speed up computation in Pitman-Yor process mixture models and hierarchical Pitman-Yor process models with no deterioration in performance, and show good results across a range of dataset sizes and data dimensionalities.

2 BACKGROUND

In this section, we will review the Pitman-Yor process and the hierarchical Pitman-Yor process, and discuss existing approaches for parallelization in Bayesian nonparametric models.

2.1 THE PITMAN-YOR PROCESS

The Dirichlet process (Ferguson, 1973) is a distribution over probability measures of the form $D := \sum_{k=1}^{\infty} \pi_k \delta_{\phi_k}$, parametrized by a concentration parameter $\alpha > 0$ and a probability measure H . The order statistics of the atom sizes π_k are described by the following stick-breaking distribution:

$$\begin{aligned} \pi_k &= w_k \prod_{j=1}^{k-1} (1 - w_j) \\ w_j &\sim \text{Beta}(1, \alpha) \end{aligned} \quad (1)$$

and the atom locations ϕ_k are sampled i.i.d. from H . The resulting probability measure D can be used to cluster observations; a finite number of observations will belong to a finite (but random) number of clusters.

The Pitman-Yor process (Perman et al., 1992; Pitman and Yor, 1997) is a two-parameter extension of the Dirichlet process, parametrized by a discount parameter $0 \leq d \leq 1$, a concentration parameter $\alpha > -d$, and a probability measure H . When the discount parameter is zero, we recover the Dirichlet process. As the discount parameter increases, we get increasingly heavy-tailed distributions over the atom sizes in the resulting probability measure. We can see this behavior by considering the stick-breaking process for the Pitman-Yor process:

$$\begin{aligned} \pi_k &= w_k \prod_{j=1}^{k-1} (1 - w_j) \\ w_j &\sim \text{Beta}(1 - d, \alpha + jd). \end{aligned} \quad (2)$$

As d increases, the rate of decay of the ordered atom sizes will decrease. When $d = 0$, we recover the stick-breaking construction for the Dirichlet process given in Equation 2. This behavior makes the Pitman-Yor process particularly appropriate for applications in language modeling. Natural language has long been known to exhibit power-law behavior (Zipf, 1935), and the Pitman-Yor process is able to capture this (Teh, 2006a).

We can use the Pitman-Yor process to cluster data using the following mixture model:

$$D \sim \text{PY}(\alpha, d, H) \quad \theta_i | D \sim D \quad x_i | \theta_i \sim f(\theta_i). \quad (3)$$

We can also construct a hierarchy of Pitman-Yor processes (Teh, 2006a) that allows us to jointly cluster multiple related groups of data. Each group is associated with a

Pitman-Yor process-distributed random measure, and the group-specific Pitman-Yor processes are coupled via a shared, Pitman-Yor process-distributed base measure. For M groups, each containing N_m data points, the generative process is

$$\begin{aligned} D_0 &\sim \text{PY}(\alpha, d, H) \\ D_m | D_0 &\sim \text{PY}(\gamma, c, D_0), \quad m = 1, \dots, M \\ \theta_{mi} | D_m &\sim D_m, \quad i = 1, \dots, N_m \\ x_{mi} | \theta_{mi} &\sim f(\theta_{mi}). \end{aligned} \quad (4)$$

This distribution has found a number of applications in text and language modeling (Teh, 2006b; Wood et al., 2009; Blunsom and Cohn, 2011).

2.2 PARALLEL METHODS FOR BAYESIAN NONPARAMETRICS

Bayesian nonparametric models allow an unbounded number of parameters, and can increase the number of parameters used as we see more data. This makes them appealing for large, complex, and potentially growing datasets. Unfortunately, naive implementation of Gibbs samplers, such as those developed in Ishwaran and James (2001), Neal (1998) and Teh et al. (2006), do not scale well to such datasets.

To counter issues of scalability, a number of authors have attempted to parallelize inference in Bayesian nonparametric models. Such algorithms typically rely on *data parallelization* – data is split onto multiple processors, and messages are passed between processors. Often, this involves making approximations that break long-range dependencies. For example in Asuncion et al. (2008) and Doshi-Velez et al. (2009), each processor maintains local sufficient statistics for the data stored on it, and approximates the full sufficient statistics by combining the local statistics with snapshots of the local statistics from other processors.

Such an approach typically leads to inaccuracies in the estimates of the global parameters or sufficient statistics. This is particularly true in Bayesian nonparametric models, where we have many components with a small number of observations. Combining the local statistics for these components is difficult, and Williamson et al. (2013) show that this leads to estimate deterioration in the case of Dirichlet processes and hierarchical Dirichlet processes. In models with power law behavior, this effect is likely to be more pronounced, due to the larger number of components with very few associated data points.

An alternative approach is to explicitly partition the model into sub-models that are independent or conditionally independent. Inference is performed on each sub-model independently, and the results are combined globally. We call

such algorithms *model-parallel*. Such models are typically also data-parallel, with different sub-models governing different subsets of the data. They also have the advantage that the sub-models typically have a smaller space of latent parameters than the full model.

Recent examples of algorithms that are both data-parallel and model-parallel are given by Williamson et al. (2013) and Lovell et al. (2012), who use auxiliary variable model representations for Dirichlet processes and hierarchical Dirichlet processes to obtain conditional independence. These algorithms hinge on the fact that we can write a Dirichlet process mixture model as a mixture of Dirichlet process mixture models, as follows:

$$\begin{aligned} D_j &\sim \text{DP}(\alpha_j, H_j), \\ \phi &\sim \text{Dirichlet}(\alpha_1, \dots, \alpha_P), \\ \mu_i | \phi &\sim \phi, \\ \theta_i | \mu_i, D_1, \dots, D_P &\sim D_{\mu_i}, \\ x_i &\sim f(\theta_i). \end{aligned} \quad (5)$$

The marginal distribution over the x_i s is the equal in distribution to that obtained by the Dirichlet process mixture model

$$\begin{aligned} D &\sim \text{DP}\left(\sum_j \alpha_j, \frac{\sum_j \alpha_j H_j}{\sum_j \alpha_j}\right), \\ \theta_i | D &\sim D, \\ x_i | \theta_i &\sim f(\theta_i). \end{aligned} \quad (6)$$

Conditioned on the μ_i s in Equation 5, we can split our model into conditionally independent sub-models involving disjoint subsets of the data, achieving both model- and data-parallelization.

3 AUXILIARY VARIABLE REPRESENTATIONS

In this section, we introduce new representations for the Pitman-Yor process and hierarchical Pitman-Yor process, that will allow us to develop model- and data-parallel inference algorithms.

3.1 AUXILIARY VARIABLE REPRESENTATION FOR THE PITMAN-YOR PROCESS

To obtain an auxiliary variable representation, we first show that a Pitman-Yor mixture model with positive concentration parameter α and continuous base measure H can be constructed as a finite mixture of Pitman-Yor mixture models. We start with a little-used representation of the atom sizes of the Pitman-Yor process.

Theorem 1 (Mixture model representation of a Pitman-Yor process). *Let $G_0 := \sum_k \rho_k \delta_{\theta_k} \sim \text{DP}(\alpha, H_0)$, and let*

$G_k := \sum_j \pi_{j,k} \delta_{\phi_{j,k}} \stackrel{i.i.d.}{\sim} \text{PY}(0, d, H)$, where H is a continuous probability measure (note that this is a normalized stable process with stable parameter d). Then $D = \sum_k \rho_k G_k$ is distributed according to a Pitman-Yor process with concentration parameter α , discount parameter d , and base measure H .

Proof. This is a direct consequence of Proposition 22 in Pitman and Yor (1997). \square

By extension, we can express a Pitman-Yor mixture model as a Dirichlet process mixture of normalized stable process mixture models, provided the concentration parameter α of the Pitman-Yor process is strictly positive and the base measure H is continuous.

Corollary 1. *The marginal distribution over the data $(x_i, i = 1, \dots, N)$ implied by the generative procedure*

$$\begin{aligned} G &\sim \text{GEM}(\alpha) \\ D_j &\sim \text{PY}(d, 0, H) \\ t_i | G &\sim G \\ \theta_i | t_i, D_1, D_2, \dots &\sim D_{t_i} \\ x_i | \theta_i &\sim f(\theta_i) \end{aligned} \quad (7)$$

is the same as the marginal distribution over the x_i obtained using the Pitman-Yor mixture model of Equation 3.

Proof. The proof is a straightforward extension of Theorem 1. \square

We have therefore reduced a Pitman-Yor mixture model with concentration parameter $\alpha > 0$ to a Dirichlet process mixture model. This allows us to apply Equation 5 and write our Pitman-Yor mixture model as a finite Dirichlet mixture of Pitman-Yor mixture models, providing the conditional independence required to construct a model-parallel sampler.

Theorem 2 (Auxiliary variable representation for Pitman-Yor mixture models). *Provided the concentration parameter $\alpha > 0$ and the base probability measure H is continuous, we can rewrite the generative process for the Pitman-Yor mixture model given in Equation 3 as:*

$$\begin{aligned} D_j &\sim \text{PY}\left(\frac{\alpha}{P}, d, H\right), \\ \phi &\sim \text{Dirichlet}\left(\frac{\alpha}{P}, \dots, \frac{\alpha}{P}\right), \\ \mu_i | \phi &\sim \phi, \\ \theta_i | \mu_i, D_1, \dots, D_P &\sim D_{\mu_i}, \\ x_i | \theta_i &\sim f(\theta_i), \end{aligned} \quad (8)$$

for $j = 1, \dots, P$ and $i = 1, \dots, N$. The marginal distribution over the x_i remains the same.

Proof. Since we can write the Pitman-Yor mixture model as a Dirichlet process mixture model, this follows as a direct application of Equation 5. An alternative proof is given in the supplement. \square

3.2 AUXILIARY VARIABLE REPRESENTATION FOR THE HIERARCHICAL PITMAN-YOR PROCESS

The results in Section 3.1 can be extended to certain special cases of the hierarchical Pitman-Yor process described in Equation 4. Unfortunately, we can only apply Theorem 1 and Corollary 1 when the base measure of the Pitman-Yor process is continuous. For the group-level Pitman-Yor processes in Equation 4, this is not the case.

The auxiliary variable representation for the Dirichlet process given in Equation 5, however, does not require a continuous base measure. We note that the Dirichlet process is a special case of the Pitman-Yor process, with discount parameter $d = 0$. We therefore work with the following special case of the hierarchical Pitman-Yor process:

$$\begin{aligned} D_0 &\sim \text{PY}(\alpha, d, H) \\ \gamma &\sim \text{Gamma}(\alpha) \\ D_m | D_0 &\sim \text{DP}(\gamma, D_0), \quad m = 1, \dots, M \\ \theta_{mi} | D_m &\sim D_m, \quad i = 1, \dots, N_m \\ x_{mi} | \theta_{mi} &\sim f(\theta_{mi}). \end{aligned} \quad (9)$$

We will refer to this construction as a hierarchical Pitman-Yor/Dirichlet process (HPY/DP). The use of a gamma dependence between the concentration parameters was first introduced by Williamson et al. (2013) in the context of the hierarchical Dirichlet process.

In Section 5, we investigate the performance of this special case of the hierarchical Pitman-Yor process on a text corpus. We find that it performs nearly as well as the more general model of Equation 4, and out-performs the hierarchical Dirichlet process (Teh et al., 2006). We therefore propose this model for large-scale text data, since it allows scalable parallel inference without significant deterioration in performance.

Theorem 3 extends the auxiliary variable representation of Theorem 2 to the hierarchical model of Equation 9.

Theorem 3 (Auxiliary variable representation for the hierarchical Pitman-Yor process). *We can rewrite the genera-*

tive process for the hierarchical model of Equation 9 as:

$$\begin{aligned} \zeta_j &\sim \text{Gamma}(\alpha/P), \\ D_{0j} &\sim \text{PY}(\alpha/P, d, H), \\ \nu_m &\sim \text{Dirichlet}(\zeta_1, \dots, \zeta_P), \\ D_{mj} | D_{0j} &\sim \text{DP}(\zeta_j, D_{0j}), \\ \mu_{mi} | \nu_m &\sim \nu_m \\ \theta_{mi} | \mu_{mi}, D_{m1}, \dots, D_{mP} &\sim D_{m\mu_{mi}} \\ x_{mi} | \theta_{mi} &\sim f(\theta_{mi}), \end{aligned} \quad (10)$$

for $j = 1, \dots, P$, $m = 1, \dots, M$, and $i = 1, \dots, N_m$. The marginal distribution over the x_i remains the same as in Equation 9.

Proof. Let $\gamma := \sum_j \zeta_j$. The normalized vector $\frac{\zeta_1, \dots, \zeta_P}{\gamma}$ is distributed according to Dirichlet $(\frac{\alpha}{P}, \dots, \frac{\alpha}{P})$, so from Theorem 1 we find that

$$D_0 := \sum_{j=1}^P \frac{\zeta_j}{\gamma} D_{0j} \sim \text{PY}(\alpha, d, H).$$

Now, for $m = 1, \dots, M$ and $j = 1, \dots, P$, let $\eta_{mj} \sim \text{Gamma}(\zeta_j)$ and $D_{mj} \sim \text{DP}(\zeta_j, D_{0j})$. The normalized vector $(\eta_{m1}, \dots, \eta_{mP}) / \sum_{j=1}^P \eta_{mj}$ is therefore distributed according to Dirichlet $(\zeta_1, \dots, \zeta_P)$. From Equation 5, we see that

$$D_m := \sum_{j=1}^P \eta_{mj} D_{mj} \sim \text{DP}(\gamma, D_0).$$

\square

The representation in Theorem 3 provides the conditional independence structure required to construct a data- and model-parallel inference algorithm.

4 INFERENCE

The auxiliary variable representation introduced in Theorem 2 makes the cluster allocations for data points $\{x_i : \mu_i = j\}$ conditionally independent of the cluster allocations for data points $\{x_i : \mu_i \neq j\}$. A similar conditional independence relationship for the hierarchical model is implied by Theorem 3. We can therefore split the data onto P parallel processors or cores, based on the values of μ_i (or μ_{mi} in the hierarchical case). We will henceforth call μ_i (μ_{mi}) the “processor indicator” for the i th data point (i th data point in the m th group).

The resulting samplers allow both model and data parallelization. Inference in Pitman-Yor mixture models and hierarchical Pitman-Yor processes scales with both the number of data points and the number of clusters. Since each

conditionally-independent sub-model only uses a subset of the data points and of the clusters, we are able to obtain significant computational advantage, as we will show empirically in Section 5.

4.1 PARALLEL INFERENCE IN THE PITMAN-YOR PROCESS

We consider first the Pitman-Yor mixture model of Equation 3. Under the auxiliary variable representation of Equation 8, each data point x_i is associated with a processor indicator μ_i and parameter θ_i . We introduce cluster indicator variables z_i , such that $z_i = z_j$ iff $\theta_i = \theta_j$. Provided the base measure H is continuous, all data points associated with a single cluster will have the same processor indicator, meaning that we can assign each cluster to one of the P processors (i.e., all data points in a single cluster are assigned to the same processor). Note that the j th processor will typically be associated with multiple clusters, corresponding to the local Pitman-Yor process D_j . Conditioned on the assignments of the processor indicators μ_i , the data points x_i in Equation 8 depend only on the local Pitman-Yor process D_{μ_i} and the associated parameters.

We can easily marginalize out the D_j and ϕ . Assume that each data point x_i is assigned to a processor $\mu_i \in \{1, \dots, P\}$, and a cluster z_i residing on that processor. We will perform *local* inference on the cluster assignments z_i , and intermittently we will perform *global* inference on the μ_i .

4.1.1 Local inference: Sampling the z_i

Conditioned on the processor assignments, the distribution over cluster assignments z_i is given by

$$P(z_i = k | \{z_j : j \neq i, \mu_j = \mu_i\}, x_i, \text{rest}) \propto \begin{cases} \frac{n_{\mu_i, k} - d}{\alpha + n_{\mu_i, \cdot}} f_k(x_i) & \text{for existing cluster } k \\ \frac{\alpha + Kd}{\alpha + n_{\mu_i, \cdot}} f^*(x_i) & \text{new cluster } k \end{cases}$$

where $n_{j, k}$ is the number of data points in the k th cluster on processor j , $f_k(x)$ is the likelihood of data point x for the k th cluster, and $f^*(x)$ is the likelihood of data point x under a new cluster.

4.1.2 Global inference: Sampling the μ_i

Under the auxiliary variable scheme, each cluster is associated with a single processor. We jointly resample the processor allocations of all data points within a given cluster, allowing us to move an entire cluster from one processor to another. We use a Metropolis Hastings step with a proposal distribution $Q(k, j_1, j_2)$ that independently assigns cluster k from processor j_1 to processor j_2 . We discuss choices of proposal distribution $Q(k, j_1, j_2)$ in Section 4.3.

The accept/reject probability is given by $r \cdot \frac{Q(k, j_2, j_1)}{Q(k, j_1, j_2)}$ where r is the likelihood ratio

$$r = \prod_{j=1}^P \frac{\Gamma(N_j^* + \alpha/P) (\alpha/P)^{(d; K_j^* - 1)}}{\Gamma(N_j + \alpha/P) (\alpha/P)^{(d; K_j - 1)}} \frac{(\alpha/P + 1 - d)^{(1; N_j - 1)}}{(\alpha/P + 1 - d)^{(1; N_j^* - 1)}} \prod_{i=1}^{\max(N_j, N_j^*)} [(1 - d)^{(1; i - 1)}]^{(a_{ij}^* - a_{ij})} \frac{a_{ij}!}{a_{ij}^*!}, \quad (11)$$

where N_j is the number of data points on processor j , a_{ij} is the number of clusters of size i on processor j and

$$(a)^{(b; c)} = \begin{cases} 1 & \text{if } c = 0 \\ a(a + b) \dots (a + (c - 1)b) & \text{for } c = 1, 2, \dots \end{cases}$$

A derivation of Equation 11 is given in the supplement. In fact, we can simplify Equation 11 further, since many of the terms in the ratio of factorials will cancel.

The reassignment of clusters can be implemented in a number of different manners. Actually transferring data from one processor to another will lead to bottlenecks, but may be appropriate if the entire data set is too large to be stored in memory on a single machine. If we can store a copy of the dataset on each machine, or we are using multiple cores on a single machine, we can simply transfer updates to lists of which data points belong to which cluster on which machine. We note that the reassignments need not occur at the same time, reducing the bandwidth required.

4.2 PARALLEL INFERENCE IN THE HIERARCHICAL PITMAN-YOR/DIRICHLET PROCESS

Again, we can assign tokens x_{mi} to one of P processors according to μ_{mi} . Conditioned on the processor assignment and the values of ζ_j , the data on each processor is distributed according to an HPY/DP. We instantiate the processor allocations μ_{mi} and the bottom-level DP parameters, plus sufficient representation to perform inference in the processor-specific HPY/DPs. We assume a Chinese restaurant franchise representation (Teh et al., 2006) – each group is represented using a “restaurant”; data points in the lower-level Dirichlet processes are clustered into “tables”; in the upper-level Pitman-Yor process, these “tables” are clustered and each cluster is assigned a “dish”.

4.2.1 Local inference: Sampling the table and dish allocations

Conditioned on the processor assignments, we simply have P independent HPY/DPs, and can use any existing inference algorithm for the hierarchical Pitman-Yor process. In our experiments, we used the Chinese restaurant franchise

sampling scheme (Teh et al., 2006; Teh, 2006a); other representations could also be used.

4.2.2 Global inference: Sampling the μ_{mi} and the ζ_j

We can represent the ζ_j as $\zeta_j := \gamma \xi_j$, where $\gamma \sim \text{Gamma}(\alpha, 1)$ and $\xi := (\xi_1, \dots, \xi_P) \sim \text{Dirichlet}(\alpha/P, \dots, \alpha/P)$. We sample ξ and the μ_{mi} jointly, and then sample γ , in order to improve the acceptance ratio of our Metropolis Hastings steps.

Again, we want to reallocate whole clusters rather than independently reallocate individual tokens. So, our proposal distribution again assigns cluster k from processor j_1 to processor j_2 with probability $Q(k, j_1, j_2)$. Note that this means that a single data point does not necessarily reside on a single processor – its tokens may be split among multiple processors. We also propose $\xi^* \sim \text{Dirichlet}(\alpha/P, \dots, \alpha/P)$, and accept the resulting state with probability $\min(1, r^{Q(k, j_1, j_2)})$, where

$$r = \prod_{j=1}^P \frac{(\xi_j^*)^{(T_j^* + \alpha/P)} T_j^*! (\alpha/P)^{(d; U_j^* - 1)}}{((\xi_j)^{(T_j + \alpha/P)}) T_j! (\alpha/P)^{(d; U_j - 1)}} \cdot \frac{(\alpha/P + 1 - d)^{(1; T_j - 1)}}{(\alpha/P + 1 - d)^{(1; T_j^* - 1)}} \cdot \left\{ \prod_{i=1}^{\max(T_j, T_j^*)} [(1 - d)^{(1; i - 1)}]^{b_{ji}^* - b_{ji}} \frac{b_{ji}!}{b_{ji}^*!} \right\} \cdot \prod_{m=1}^M \prod_{i=1}^{\max(N_j, N_j^*)} \frac{a_{jmi}!}{a_{jmi}^*!}. \quad (12)$$

Here, T_{mj} is the total number of occupied tables from the m th restaurant on processor j , U_j is the total number of unique dishes on processor j , a_{jmi} is the total number of tables in restaurant m on processor j with exactly i customers, and b_{ji} is the total number of dishes on processor j served at exactly i tables. Many of the ratios can be simplified further, reducing computational costs. A derivation of Equation 12 is given in the supplement.

As with the sampler described in Section 4.1, we can either transfer the data between machines, or simply update lists of which data points are “active” on each machine. We can resample γ after sampling ξ and the μ_{mi} using a standard Metropolis Hastings step.

4.3 CHOICE OF PROPOSAL DISTRIBUTION

There are many valid choices for the proposal distributions $Q(k, j_1, j_2)$ used to sample the μ_i and μ_{mi} in Sections 4.1 and 4.2. We tried several different proposal distributions and found we obtained good mixing when

$$Q(k, j_1, j_2) = \begin{cases} \frac{1}{P-1} s_{j_1} & \text{if } j_1 \neq j_2 \\ 1 - s_{j_1} & \text{if } j_1 = j_2 \end{cases} \quad (13)$$

where s_{j_1} is the fraction of data in processor j_1 . As discussed by Gal and Ghahramani (2013), due to cluster size imbalance inherent to the Pitman-Yor process, we do not expect to see even load balance; however this proposal distribution encourages processors with a larger proportion of the data (higher load) to transfer data to under-used data. Since at any point the fraction of points in any cluster is small it also reduces the number of transfer and hence network load. We will show in Section 5 that this not only gives good performance but also gives good load sharing among multiple processors.

5 EVALUATION

We expect the inference algorithms presented in Section 4 to yield faster inference than a non-parallel implementation. Each processor is locally performing inference in a Pitman-Yor mixture model or a hierarchical Pitman-Yor/Dirichlet process. These models only contain a subset of the total number of clusters and of the total data set. Since inference in these models scales at least linearly (depending on likelihood) with both the number of data points and the number of latent components, we expect them to converge much more quickly than a Pitman-Yor mixture model or hierarchical Pitman-Yor/Dirichlet process on the entire data set. Provided the computational cost of the global steps remains relatively low, and provided the global steps achieve sufficiently fast mixing, we expect to see overall speed-ups. Further, we expect the estimate quality to remain high, since our algorithms do not introduce approximations.

In both cases, we evaluate via comparison with non-parallel implementations of the model. We are unaware of any other parallelizable inference algorithms for the Pitman-Yor process and its extensions.

5.1 PITMAN-YOR MIXTURE OF GAUSSIANS

We first evaluate performance of the parallel sampler for the Pitman-Yor mixture model, described in Section 4.1, using synthetic data. We generated data sets of varying size N and data dimensionality D , to evaluate performance on different sized datasets. Each data set contained $N/2000$ clusters of size 500, $N/4000$ clusters of size 1000, $N/10000$ clusters of size 2500, and $N/20000$ clusters of size 5000, giving $K = 9N/10000$ clusters in total. This was designed to give many smaller clusters and fewer larger clusters. Each cluster parametrized a univariate Gaussian with unit variance and mean sampled according to a $\text{Uniform}((-K/2, K/2)^D)$ distribution.

We ran our algorithm on 90% of the resulting data sets using the algorithm described in Section 4.1, on 1, 2, 4 and 8 cores of a multi-core machine. Each algorithm was initialized by clustering the data into 80 clusters. We performed

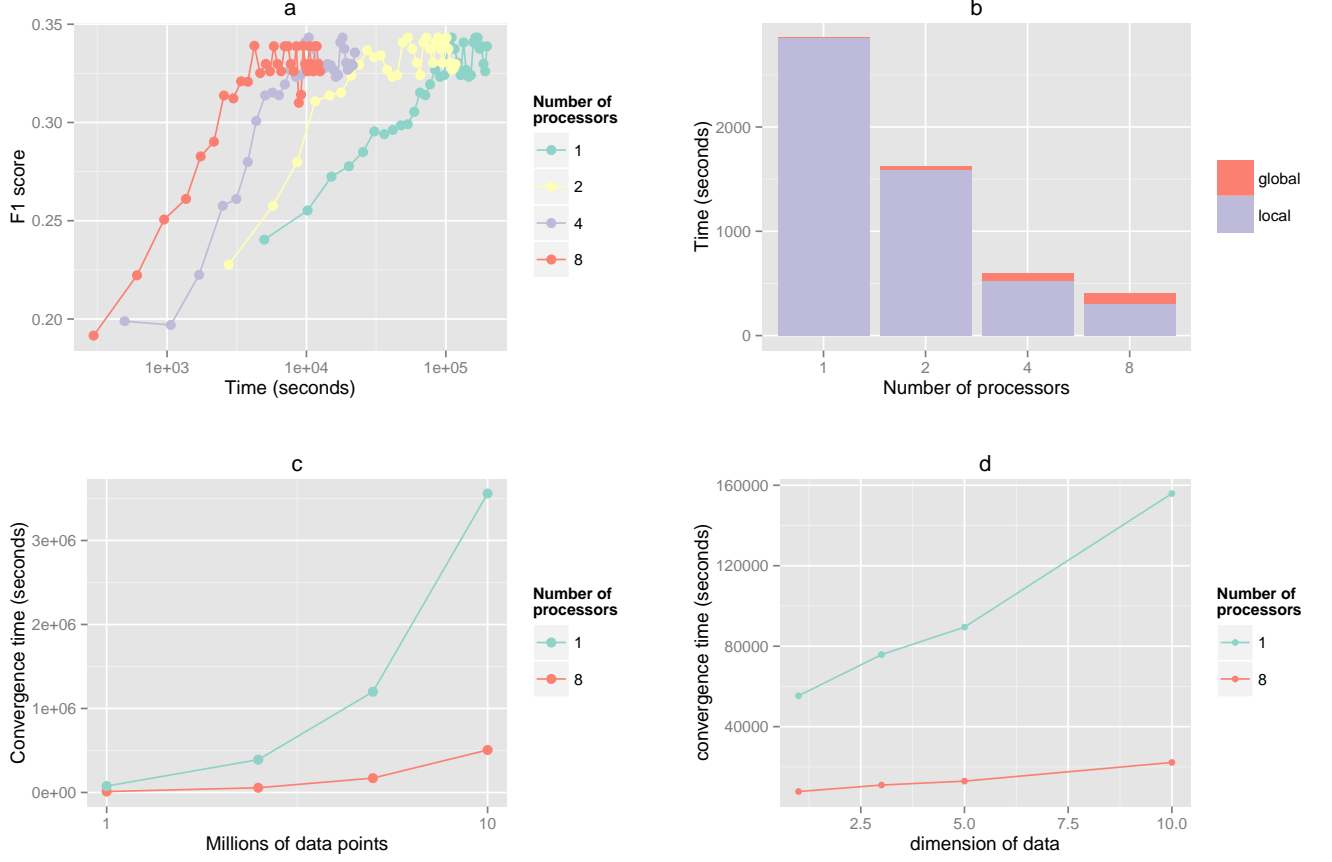


Figure 1: Evaluation on synthetic data modeled using a Pitman-Yor mixture model. a: F1 score vs run time; b: Amount of time spent on global vs local computation; c: Time taken to reach convergence ($< 0.1\%$ change in training set log likelihood) vs number of data points; d: Time taken to reach convergence vs data dimension.

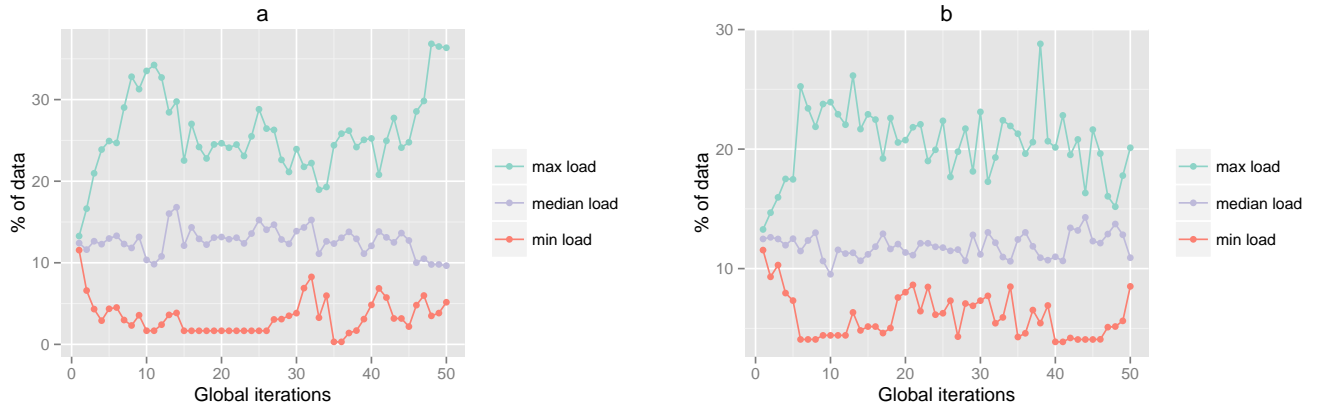


Figure 2: Maximum, median, and minimum loads per global iteration, using (a) a uniform proposal distribution; (b) the proposal distribution given in Equation 13.

100 iterations of the local inference step for each iteration of the global inference step. The concentration parameter α was set to 0.5 and d was set to 0.1; these values were selected via grid search. We evaluated by calculating the test set log likelihood for the remaining 10% of the data.

Figure 1(a) shows how the log likelihood of the test sample varies with time, for one million datapoints with $D = 3$. We see that we get good speedup by increasing the number of processors, while converging to (approximately) the same value for all experiments. Figure 1(b) shows that the amount of time spent on local computation far exceeds that

Data Size	1M	2.5M	5M	10 M
Efficiency	0.864	0.873	0.877	0.879
Dimension	1	3	5	10
Efficiency	0.893	0.864	0.866	0.877
Processors		2	4	8
Efficiency		0.880	0.865	0.864

Table 1: Efficiency with varying data size (with $D = 3$ and $P = 8$), varying data dimension (with $N = 1M$ $P = 8$) and number of processor (with $N = 1M$ and $D = 3$).

spent on global steps, explaining why we have a faster per-iteration time. Figures 1(c) and 1(d) show that the decrease in computational speed is apparent at different sizes of data set N and data dimensionality D .

Following Asuncion et al. (2008), we report the efficiency of our model. If a model running on P processor converges in time T_p while the single processor model converges in time T then the efficiency is calculated as $\frac{T}{P \cdot T_p}$. Table 1 shows how efficiency varies if we change the number of data points, the dimensionality of each data point, and the number of processors. An efficiency of 1 would indicate a linear speed-up – using P processors is P times as fast as using one processor. We get efficiency very close to the linear speedup as shown in Table 1.

Next we evaluate how evenly computation is split between the cores. Figure 2 shows the how the data is split between cores over time. Figure 2(b) shows the load distribution obtained using the proposal distribution of Equation 13, and Figure 2(a) shows the load distribution obtained using the uniform distribution used by Williamson et al. (2013). The maximum load governs the amount of time spend performing local computation (which was seen in Figure 1(b) to dominate the computation time). While, as we would expect (Gal and Ghahramani, 2013), we have uneven loads in both cases, we achieve better load balancing using the new proposal.

5.2 HIERARCHICAL PITMAN-YOR/DIRICHLET PROCESS

In this section, we evaluate the sampler described in Section 4.2 on two text data sets:

- NIPS¹: A collection of 2470 papers from the NIPS conference, collected between 1988 and 2003 which includes 14300 unique words and a total of 3, 280, 697 words.
- ENRON²: A collection of 39861 emails including 28102 unique words and a total of 6, 400, 000 words.

¹<http://ai.stanford.edu/~gal/data.html>

²<https://archive.ics.uci.edu/ml/datasets/Bag+of+Words>

Dataset	HDP	HPY/DP	HPY
NIPS	1706.52	1650.44	1621.34
ENRON	2110.98	2054.85	2018.36

Table 2: Test set perplexity for different models.

Processors	2	4	8
NIPS	0.855	0.805	0.824
ENRON	0.854	0.807	0.817

Table 3: Efficiency of the HPY/DP algorithm with varying number of processors.

In each case, we held out 10% of the data for testing, and evaluated our algorithms using perplexity on the held out test set, as calculated in Asuncion et al. (2008).

We begin by considering how much performance is gaining by restricting the lower-level stochastic processes to be Dirichlet processes, rather than Pitman-Yor processes. Table 2 compares the full hierarchical Pitman-Yor process described in Equation 4 (denoted HPY), the model implemented using our algorithm and described in Equation 9 (denoted HPY/DP), and the hierarchical Dirichlet process Teh et al. (2006) (denoted HDP).

We find that, while the full hierarchical Pitman-Yor process obtains the best perplexity, the HPY/DP model still performs better than the hierarchical Dirichlet process. Since there is not, currently, a scalable inference algorithm for the full hierarchical Pitman-Yor process, we argue that the proposed algorithm and model offer a good trade-off between scalability and performance

Having established the applicability of the model, we consider scalability. Figures 3(a) and 3(b) show how the sample test set perplexity changes with time using 1,2,4 and 8 processors on the NIPS and ENRON data sets, respectively. As with the Pitman-Yor mixture model, we see that increasing the number of processors yields improvements in computation time. Figures 3(c) and 3(d) show that, as before, this occurs because the cost of the local computations decreases as we add more processors, and remains high relative to the cost of the global computations. This is reflected in the efficiencies obtained for different numbers of processors (Table 3), which remain close to one.

6 DISCUSSION AND FUTURE WORK

In this paper, we have presented new auxiliary variable representations for certain cases of the Pitman-Yor process and the hierarchical Pitman-Yor process. These representations allowed us to make use of conditional independencies to develop inference schemes that are both data- and model-parallel.

While this paper provides a significant step forward in

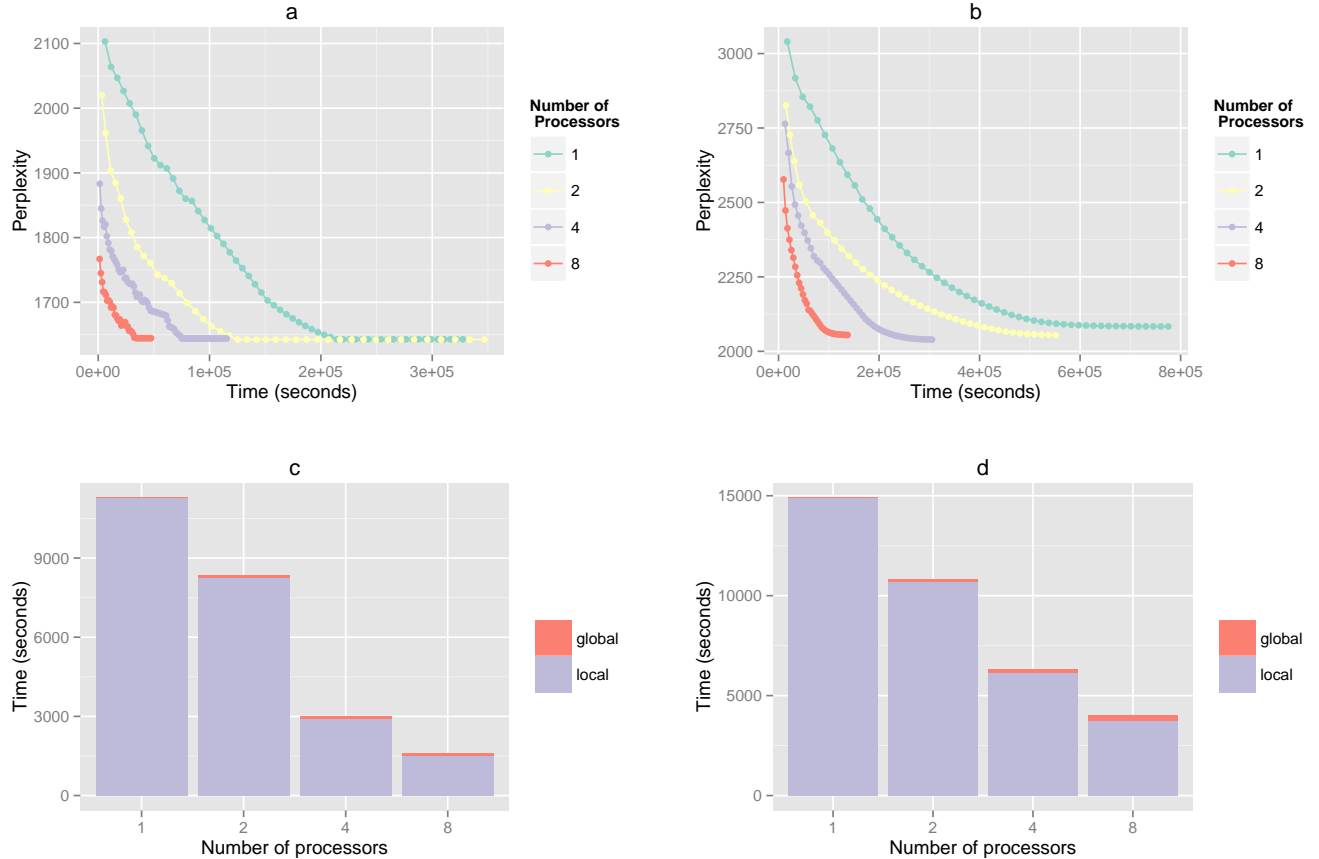


Figure 3: Evaluation on text corpora using the HPY/DP model. a: Test set perplexity vs run time (NIPS); b: Test set perplexity vs run time (ENRON); c: Amount of time spent on global vs local computation (NIPS); d: Amount of time spent on global vs local communication (ENRON).

the development of parallel inference algorithms for the Pitman-Yor process, it does not cover all algorithms of interest. The auxiliary variable representation introduced in Theorem 2 requires a positive concentration parameter. It remains an open question whether there exist alternative representations for $\alpha < 0$ that yield the desired conditional independence structure. Further, our auxiliary variable representation requires a continuous base measure H . While this is typically the case for Pitman-Yor mixture models, it is not the case for the more general hierarchical Pitman-Yor process described in Equation 4. We hope that this work inspires further research into scalable inference for models beyond the Dirichlet process, allowing parallel algorithms for this and other models.

Acknowledgements

This research was supported by *NSF Social IIS1111142*, *DARPA XDATA FA87501220324* and *NIH GWAS R01GM087694*.

References

- Asuncion, A., Smyth, P., and Welling, M. (2008). Asynchronous distributed learning of topic models. In *Advances in Neural Information Processing Systems*.
- Blunsom, P. and Cohn, T. (2011). A hierarchical Pitman-Yor process HMM for unsupervised part of speech induction. In *Association for Computational Linguistics: Human Language Technologies (HLT)*.
- Chang, J. and Fisher III, J. W. (2013). Parallel sampling of DP mixture models using sub-clusters splits. In *Advances in Neural Information Processing Systems*.
- Doshi-Velez, F., Knowles, D., Mohamed, S., and Ghahramani, Z. (2009). Large scale nonparametric Bayesian inference: Data parallelisation in the Indian buffet process. In *Advances in Neural Information Processing Systems*.
- Ferguson, T. S. (1973). A Bayesian analysis of some non-parametric problems. *Annals of Statistics*, 1(2):209–230.
- Gal, Y. and Ghahramani, Z. (2013). Pitfalls in the use of parallel inference for the Dirichlet process. In *Workshop in Big Learning, NIPS*.

- Goldwater, S., Griffiths, T., and Johnson, M. (2006). Interpolating between types and tokens by estimating power-law generators. In *Advances in Neural Information Processing Systems*.
- Ishwaran, H. and James, L. (2001). Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, 96(453):161–173.
- Lovell, D., Adams, R., and Mansinghka, V. (2012). Parallel Markov chain Monte Carlo for Dirichlet process mixtures. In *Workshop on Big Learning, NIPS*.
- Neal, R. (1998). Markov chain sampling methods for Dirichlet process mixture models. Technical Report 9815, Dept. of Statistics, University of Toronto.
- Perman, M., Pitman, J., and Yor, M. (1992). Size-biased sampling of Poisson point processes and excursions. *Probability Theory and Related Fields*, 92(1):21–39.
- Pitman, J. and Yor, M. (1997). The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *Annals of Probability*, 25(2):531–1010.
- Teh, Y.-W. (2006a). A Bayesian interpretation of interpolated Keyser-Ney. Technical Report TAR2/06, National University of Singapore.
- Teh, Y.-W. (2006b). A hierarchical Bayesian language model based on Pitman-Yor processes. In *ACL*.
- Teh, Y.-W., Jordan, M., Beal, M., and Blei, D. (2006). Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.
- Williamson, S., Dubey, A., and Xing, E. (2013). Parallel Markov chain Monte Carlo for nonparametric mixture models. In *International Conference on Machine Learning*.
- Wood, F., Archambeau, C., Gasthaus, J., James, L., and Teh, Y.-W. (2009). A stochastic memoizer for sequence data. In *International Conference on Machine Learning*.
- Zipf, G. (1935). *The Psychobiology of Language*. Houghton-Mifflin.

Market Making with Decreasing Utility for Information

Miroslav Dudík
Microsoft Research

Rafael Frongillo
Microsoft Research

Jennifer Wortman Vaughan
Microsoft Research

Abstract

We study information elicitation in cost-function-based combinatorial prediction markets when the market maker’s utility for information decreases over time. In the *sudden revelation* setting, it is known that some piece of information will be revealed to traders, and the market maker wishes to prevent guaranteed profits for trading on the sure information. In the *gradual decrease* setting, the market maker’s utility for (partial) information decreases continuously over time. We design adaptive cost functions for both settings which: (1) preserve the information previously gathered in the market; (2) eliminate (or diminish) rewards to traders for the publicly revealed information; (3) leave the reward structure unaffected for other information; and (4) maintain the market maker’s worst-case loss. Our constructions utilize mixed Bregman divergence, which matches our notion of utility for information.

1 INTRODUCTION

Prediction markets have been used to elicit information in a variety of domains, including business [6, 7, 12, 28], politics [4, 29], and entertainment [25]. In a prediction market, traders buy and sell *securities* with values that depend on some unknown future outcome. For example, a market might offer securities worth \$1 if Norway wins a gold medal in Men’s Moguls in the 2014 Winter Olympics and \$0 otherwise. Traders are given an incentive to reveal their beliefs about the outcome by buying and selling securities, e.g., if the current price of the above security is \$0.15, traders who believe that the probability of Norway winning is more than 15% are incentivized to buy and those who believe that the probability is less than 15% are incentivized to sell. The equilibrium price reflects the market consensus about the security’s expected payout (which here coincides with the probability of Norway winning the medal).

There has recently been a surge of research on the design of prediction markets operated by a centralized authority called a *market maker*, an algorithmic agent that offers to buy or sell securities at some current price that depends on the history of trades in the market. Traders in these markets can express their belief whenever it differs from the current price by either buying or selling, regardless of whether other traders are willing to act as a counterparty, because the market maker always acts as a counterparty, thus “providing the liquidity” and subsidizing the information collection. This is useful in situations when the lack of interested traders would negatively impact the efficiency in a traditional exchange. Of particular interest to us are *combinatorial prediction markets* [8–10, 17–19, 26] which offer securities on various related events such as “Norway wins a total of 4 gold medals in the 2014 Winter Olympics” and “Norway wins a gold medal in Men’s Moguls.” In combinatorial markets with large, expressive security spaces, such as an Olympics market with securities covering 88 nations participating in 98 events, the lack of an interested counterparty is a major concern. Only a single trader may be interested in trading the security associated with a specific event, but we would still like the market to incorporate this trader’s information.

Most market makers considered in the literature are implemented using a pricing function called the *cost function* [11]. While such markets have many favorable properties [1, 2], the current approaches have several drawbacks that limit their applicability in real-world settings. First, existing work implicitly assumes that the outcome is revealed all at once. When concerned about “just-in-time arbitrage,” in which traders closer to the information source make last-minute guaranteed profits by trading on the sure information before the market maker can adjust prices, the market maker can prevent such profits by closing the entire market just before the outcome is revealed. This approach is undesirable when partial information about the outcome is revealed over time, as is often the case in practice, including the Olympics market. For instance, we may learn the results of Men’s Moguls before Ladies’ Figure Skating has taken place. Closing a large combinatorial market when

ever a small portion of the outcome is determined seems to be an unreasonably large intervention.

Second, in real markets, the information captured by the market’s consensus prices often becomes less useful as the revelation of the outcome approaches. Consider a market over the event “Unemployment in the U.S. falls below 5.8% by the end of 2015.” Although there may be a particular moment when the unemployment rate is publicly revealed, this information becomes gradually less useful as that moment approaches; the government may be less able to act on the information as the end of the year draws near. In the Olympics market, the outcome of a particular competition is often more certain as the final announcement approaches, e.g., if one team is far ahead by the half-time of a hockey game, market forecasts become less interesting. Existing market makers fail to take this diminishing utility for information into account, with the strength of the market incentives remaining constant over time.

To address these two shortcomings of existing markets, we consider two settings:

- a *sudden revelation* setting in which it is known that some piece of information (such as the winner of Men’s Moguls) will be publicly revealed at a particular time, driving the market maker’s utility for this information to zero; crucially, in this setting we assume that the market maker *does not* have direct access to this information at the time it is revealed, which is realistic in the case of the Olympics where a human might not be available to input winners for all 98 events in real time;
- a *gradual decrease* setting in which the market maker has a diminishing utility for a piece of information (such as the unemployment rate for 2015) over time and therefore is increasingly unwilling to pay for this information even while other information remains valuable.

The sudden revelation setting can be viewed as a special case of the gradual decrease setting. In both cases, we model the relevant information as a variable X , representing a partly determined outcome such as the identity of the gold medal winner in a single sports event.

We consider cost-function-based market makers in which the cost function switches one or many times, and aim to design switching strategies such that: (1) information previously gathered in the market is not lost at the time of the switch, (2) a trader who knows the value of X but has no additional information is unable to profit after the switch (for the sudden revelation setting) or is able to profit less and less over time (in the gradual decrease setting), and (3) the market maker maintains the same reward structure for any other information that traders may have. To formalize these objectives, we define the notion of the market maker’s utility (Sec. 2) and show how it corresponds to the *mixed Bregman divergence* [13, 15] (Sec. 2.5).

For the sudden revelation setting (Sec. 3), we introduce a generic cost function switching technique which in many cases removes the rewards for “just-in-time arbitragers” who know only the value of X , while allowing traders with other information to profit, satisfying our objectives.

For the gradual decrease setting (Sec. 4), we focus on *linearly constrained market makers* (LCMMs) [13], proposing a time-sensitive market maker that gradually decreases liquidity by employing the cost function of a different LCMM at each point in time, again meeting our objectives.

Others have considered the design of cost-function-based markets with adaptive liquidity [3, 21–24]. That line of research has typically focused on the goal of slowing down price movement as more money enters the market. In contrast, we adjust liquidity to reflect the current market maker’s utility which can be viewed as something external to trading in the market. Additionally, we change liquidity only in the “low-utility” parts of the market, whereas previous work considered market-wide liquidity shifts. Brahma et al. [5] designed a Bayesian market maker that adapts to perceived increases in available information. Our market maker does not try to infer high information periods, but assumes that a schedule of public revelations is given a priori. Our market makers have guaranteed bounds on worst-case loss whereas those of Brahma et al. [5] do not.

2 SETTING AND DESIDERATA

We begin by reviewing cost-function-based market making before describing our desiderata. Here and throughout the paper we make use of many standard results from convex analysis, summarized in Appendix A. All of the proofs in this paper are relegated to the appendix.¹

2.1 COST-FUNCTION-BASED MARKET MAKING

Let Ω denote the *outcome space*, a finite set of mutually exclusive and exhaustive states of the world. We are interested in the design of cost-function-based market makers operating over a set of K *securities* on Ω specified by a *payoff function* $\rho : \Omega \rightarrow \mathbb{R}^K$, where $\rho(\omega)$ denotes the vector of security payoffs if the outcome $\omega \in \Omega$ occurs. Traders may purchase *bundles* $\mathbf{r} \in \mathbb{R}^K$ of securities from the market maker, with r_i denoting the quantity of security i that the trader would like to purchase; negative values of r_i are permitted and represent short selling. A trader who purchases a bundle \mathbf{r} of securities pays a specified cost for this bundle up front and receives a (possibly negative) payoff of $\rho(\omega) \cdot \mathbf{r}$ if the outcome $\omega \in \Omega$ occurs.

Following Chen and Pennock [11] and Abernethy et al. [1, 2], we assume that the market maker initially prices securities using a convex potential function $C : \mathbb{R}^K \rightarrow \mathbb{R}$,

¹The full version of this paper on arXiv includes the appendix.

called the *cost function*. The current state of the market is summarized by a vector $\mathbf{q} \in \mathbb{R}^K$, where q_i denotes the total number of shares of security i that have been bought or sold so far. If the market state is \mathbf{q} and a trader purchases the bundle \mathbf{r} , he must pay the market maker $C(\mathbf{q} + \mathbf{r}) - C(\mathbf{q})$. The new market state is then $\mathbf{q} + \mathbf{r}$. The *instantaneous price* of security i is $\partial C(\mathbf{q}) / \partial q_i$ whenever well-defined; this is the price per share of an infinitesimally small quantity of security i , and is frequently interpreted as the traders' collective belief about the expected payoff of this security. Any expected payoff must lie in the convex hull of the set $\{\rho(\omega)\}_{\omega \in \Omega}$, called *price space*, denoted \mathcal{M} .

While our cost function might not be differentiable at all states \mathbf{q} , it is always *subdifferentiable* thanks to convexity, i.e., its subdifferential $\partial C(\mathbf{q})$ is non-empty for each \mathbf{q} and, if it is a singleton, it coincides with the gradient. Let $\mathbf{p}(\mathbf{q}) := \partial C(\mathbf{q})$ be called the *price map*. The set $\mathbf{p}(\mathbf{q})$ is always convex and can be viewed as a multi-dimensional version of the "bid-ask spread". In a state \mathbf{q} , a trader can make an expected profit if and only if he believes that $\mathbb{E}[\rho(\omega)] \notin \mathbf{p}(\mathbf{q})$. If C is differentiable at \mathbf{q} , we slightly abuse notation and also use $\mathbf{p}(\mathbf{q}) := \nabla C(\mathbf{q})$.

We assume that the cost function satisfies two standard properties: *no arbitrage* and *bounded loss*. The former means that as long as all outcomes ω are possible, there are no market transactions with a guaranteed profit for a trader. The latter means that the worst-case loss of the market maker is a priori bounded by a constant. Together, they imply that the cost function C can be written in the form $C(\mathbf{q}) = \sup_{\mu \in \mathcal{M}} [\mu \cdot \mathbf{q} - R(\mu)]$, where R is the convex conjugate of C , with $\text{dom } R = \mathcal{M}$. See Abernethy et al. [1, 2] for an analysis of the properties of such markets.

Example 1. Logarithmic market-scoring rule (LMSR). The LMSR of Hanson [18, 19] is a cost function for a *complete market* where traders can express any probability distribution over Ω . Here, for any $K \geq 1$, $\Omega = [K] := \{1, \dots, K\}$ and $\rho_i(\omega) = \mathbf{1}[i = \omega]$ where $\mathbf{1}[\cdot]$ is a 0/1 indicator, i.e., the security i pays out \$1 if the outcome i occurs and \$0 otherwise. The price space \mathcal{M} is the simplex of probability distributions in K dimensions. The cost function is $C(\mathbf{q}) = \ln(\sum_{i=1}^K e^{q_i})$, which is differentiable and generates prices $p_i(\mathbf{q}) = e^{q_i} / (\sum_{j=1}^K e^{q_j})$. Here R is the negative entropy function, $R(\mu) = \sum_{i=1}^K \mu_i \ln \mu_i$.

Example 2. Square. The square market consists of two independent securities ($K = 2$) each paying out either \$0 or \$1. This can be encoded as $\Omega = \{0, 1\}^2$ with $\rho_i(\omega) = \omega_i$ for $i = 1, 2$. The price space is the unit square $\mathcal{M} = [0, 1]^2$. Consider the cost function $C(\mathbf{q}) = \ln(1 + e^{q_1}) + \ln(1 + e^{q_2})$, which is differentiable and generates prices $p_i(\mathbf{q}) = e^{q_i} / (1 + e^{q_i})$ for $i = 1, 2$. Using this cost function is equivalent to running two independent binary markets, each with an LMSR cost function. We have $R(\mu) = \sum_{i=1}^2 \mu_i \ln \mu_i + (1 - \mu_i) \ln(1 - \mu_i)$.

PROTOCOL 1: Sudden Revelation Market Makers

Input: initial cost function C , initial state \mathbf{s}^{ini} , switch time t , update functions $\text{NewCost}(\mathbf{q})$, $\text{NewState}(\mathbf{q})$

Until time t :
 sell bundles $\mathbf{r}^1, \dots, \mathbf{r}^N$ priced using C
 for the total cost $C(\mathbf{s}^{\text{ini}} + \mathbf{r}) - C(\mathbf{s}^{\text{ini}})$ where $\mathbf{r} = \sum_{i=1}^N \mathbf{r}^i$
 let $\mathbf{s} = \mathbf{s}^{\text{ini}} + \mathbf{r}$

At time t :
 $\tilde{C} \leftarrow \text{NewCost}(\mathbf{s})$
 $\tilde{\mathbf{s}} \leftarrow \text{NewState}(\mathbf{s})$

After time t :
 sell bundles $\tilde{\mathbf{r}}^1, \dots, \tilde{\mathbf{r}}^{\tilde{N}}$ priced using \tilde{C}
 for the total cost $\tilde{C}(\tilde{\mathbf{s}} + \tilde{\mathbf{r}}) - \tilde{C}(\tilde{\mathbf{s}})$ where $\tilde{\mathbf{r}} = \sum_{i=1}^{\tilde{N}} \tilde{\mathbf{r}}^i$
 let $\tilde{\mathbf{s}}^{\text{fin}} = \tilde{\mathbf{s}} + \tilde{\mathbf{r}}$

Observe ω
 Pay $(\mathbf{r} + \tilde{\mathbf{r}}) \cdot \rho(\omega)$ to traders

PROTOCOL 2: Gradual Decrease Market Makers

Input: time-sensitive cost function $\mathbf{C}(\mathbf{q}; t)$, initial state \mathbf{s}^0 , initial time t^0 , update function $\text{NewState}(\mathbf{q}; t, t')$

For $i = 1, \dots, N$ (where N is an unknown number of trades):
 at time $t^i \geq t^{i-1}$: receive a request for a bundle \mathbf{r}^i
 $\tilde{\mathbf{s}}^{i-1} \leftarrow \text{NewState}(\mathbf{s}^{i-1}; t^{i-1}, t^i)$
 sell the bundle \mathbf{r}^i
 for the cost $\mathbf{C}(\tilde{\mathbf{s}}^{i-1} + \mathbf{r}^i; t^i) - \mathbf{C}(\tilde{\mathbf{s}}^{i-1}; t^i)$
 $\mathbf{s}^i \leftarrow \tilde{\mathbf{s}}^{i-1} + \mathbf{r}^i$

Observe ω
 Pay $\sum_{i=1}^N \mathbf{r}^i \cdot \rho(\omega)$ to traders

Example 3. Piecewise linear cost. Here we describe a non-differentiable cost function for a single binary security ($K = 1$). Let $\Omega = \{0, 1\}$ and $\rho(\omega) = \omega$, so $\mathcal{M} = [0, 1]$. The cost function is $C(q) = \max\{0, q\}$. It gives rise to the price map such that $p(q) = 0$ if $q < 0$, and $p(q) = 1$ if $q > 0$, but at $q = 0$, we have $p(q) = [0, 1]$, i.e., because of non-differentiability we have a bid-ask spread at $q = 0$. Here, $R(\mu) = \mathbb{I}[\mu \in [0, 1]]$ where $\mathbb{I}[\cdot]$ is a 0/ ∞ indicator, equal to 0 if true and ∞ if false. This market is uninteresting on its own, but will be useful to us in Sec. 3.3.

2.2 OBSERVATIONS AND ADAPTIVE COSTS

We study two settings. In the *sudden revelation setting*, it is known to both the market maker and the traders that at a particular point in time (the observation time) some information about the outcome (an observation) will be publicly revealed to the traders, but not to the market maker. More precisely, let any function on Ω be called a *random variable* and its value called the *realization* of this random variable. Given a random variable $X : \Omega \rightarrow \mathcal{X}$, we assume that its realization is revealed to the traders at the observation time. For a random variable X and a possible realization x , we define the *conditional outcome space* by $\Omega^x := \{\omega \in \Omega : X(\omega) = x\}$. After observing $X = x$ (where, using standard random variable shorthand, we write X for $X(\omega)$), the traders can conclude that

$\omega \in \Omega^x$. Note that the sets $\{\Omega^x\}_{x \in \mathcal{X}}$ form a partition of Ω .

We design *sudden revelation market makers* (Protocol 1) that replace the cost function C with a new cost function \tilde{C} , and the current market state s (i.e., the current value of q in the definition above) with a new market state \tilde{s} in order to reflect the decrease in the utility for information about X . Such a switch would typically occur just before the observation time. Note that we allow the new cost function \tilde{C} as well as the new state \tilde{s} to be chosen adaptively according to the last state s of the original cost function C .

In the *gradual decrease* setting, the utility for information about a future observation X is decreasing continuously over time. We use a *gradual decrease market maker* (Protocol 2) with a time-sensitive cost function $C(q; t)$ which sells a bundle r for the cost $C(q+r; t) - C(q; t)$ at time t , when the market is in a state q . We place no assumptions on C other than that for each t , the function $C(\cdot; t)$ should be an arbitrage-free bounded-loss cost function. The market maker may modify the state between the trades.

Protocol 2 alternates between trades and cost-function switches akin to those in Protocol 1. In each iteration i , the cost function $C(\cdot; t^{i-1})$ is replaced by the cost function $C(\cdot; t^i)$ while simultaneously replacing the state s^{i-1} by the state \tilde{s}^{i-1} . Crucially, unlike Protocol 1, the cost-function switch here is *state independent*, so any state-dependent adaptation happens through the state update.²

At a high level, within each of the protocols, our goal is to design switch strategies that satisfy the following criteria:

- Any information that has already been gathered from traders about the relative likelihood of the outcomes in the conditional outcome spaces is preserved.
- A trader who has information about the observation X but has no additional information about the relative likelihood of outcomes in the conditional outcome spaces is unable to profit from this information (for sudden revelation), or the profits of such a trader are decreasing over time (for gradual decrease).
- The market maker continues to reward traders for new information about the relative likelihood of outcomes in the conditional outcome spaces as it did before, with prices reflecting the market maker's utility for information within these sets of outcomes.

To reason about these goals, it is necessary to define what we mean by the information that has been gathered in the market and the market maker's utility.

2.3 MARKET MAKER'S UTILITY

By choosing a cost function, the market maker creates an incentive structure for the traders. Ideally, this incentive

structure should be aligned with the market maker's subjective utility for information. That is, the amount the market maker is willing to pay out to traders should reflect the market maker's utility for the information that the traders have provided. In this section, we study how the traders are rewarded for various kinds of information, and use the magnitude of their profits to define the market maker's implicit "utility for information" formally.

We start by defining the market maker's utility for a belief, where a *belief* $\mu \in \mathcal{M}$ is a vector of expected security payoffs $\mathbb{E}[\rho(\omega)]$ for some distribution over Ω .

Definition 1. The market maker's utility for a belief $\mu \in \mathcal{M}$ relative to the state q is the maximum expected payoff achievable by a trader with belief μ when the current market state is q :

$$\text{Util}(\mu; q) := \sup_{r \in \mathbb{R}^K} [\mu \cdot r - C(q+r) + C(q)] .$$

Any subset $\mathcal{E} \subseteq \Omega$ is referred to as an *event*. Observations $X = x$ correspond to events Ω^x . Suppose that a trader has observed an event, i.e., a trader knows that $\omega \in \mathcal{E}$, but is otherwise uninformed. The market maker's utility for that event can then be naturally defined as follows.

Definition 2. The utility for a (non-null) event $\mathcal{E} \subseteq \Omega$ relative to the market state q is the largest guaranteed payoff that a trader who knows $\omega \in \mathcal{E}$ (and has only this information) can achieve when the current market state is q :

$$\text{Util}(\mathcal{E}; q) := \sup_{r \in \mathbb{R}^K} \min_{\omega \in \mathcal{E}} [\rho(\omega) \cdot r - C(q+r) + C(q)] .$$

Finally, consider the setting in which a trader has observed an event \mathcal{E} , and also holds a belief μ consistent with \mathcal{E} . Specifically, let $\mathcal{M}(\mathcal{E})$ denote the convex hull of $\{\rho(\omega)\}_{\omega \in \mathcal{E}}$, which is the set of beliefs consistent with the event \mathcal{E} , and assume $\mu \in \mathcal{M}(\mathcal{E})$. Then we can define the "excess utility for the belief μ " as the excess utility provided by μ over just the knowledge of \mathcal{E} .

Definition 3. Given an event \mathcal{E} and a belief $\mu \in \mathcal{M}(\mathcal{E})$, the excess utility of μ over \mathcal{E} , relative to the state q is:

$$\text{Util}(\mu \mid \mathcal{E}; q) = \text{Util}(\mu; q) - \text{Util}(\mathcal{E}; q) .$$

Note that in these definitions a trader can always choose not to trade ($r = 0$), so the utility for a belief and an event is non-negative. Also it is not too difficult to see that $\text{Util}(\mu; q) \geq \text{Util}(\mathcal{E}; q)$ for any $\mu \in \mathcal{M}(\mathcal{E})$, so the excess utility for a belief is also non-negative.

In Sec. 2.5, we show that given a state q and a non-null event \mathcal{E} , there always exists a (possibly non-unique) belief $\mu \in \mathcal{E}$ such that $\text{Util}(\mu \mid \mathcal{E}; q) = 0$. Thus, a trader with such a "worst-case" belief is able to achieve in expectation no reward beyond what any trader that just observed \mathcal{E} would receive. We show that these worst-case beliefs correspond to certain kinds of "projections" of the current price $p(q)$ onto $\mathcal{M}(\mathcal{E})$. For LMSR, the projections are with respect to KL divergence and correspond to the usual conditional probability distributions. Moreover, for sufficiently

²This simplifying restriction matches our solution concept in Sec. 4, but it could be dropped for greater generality.

Table 1: Information Desiderata

PRICE	<i>Preserve prices:</i> $\tilde{p}(\tilde{s}) = p(s).$
CONDPRICE	<i>Preserve conditional prices:</i> $\tilde{p}(X=x; \tilde{s}) = p(X=x; s) \quad \forall x \in \mathcal{X}.$
DECUTIL	<i>Decrease profits for uninformed traders:</i> $\tilde{\text{Util}}(X=x; \tilde{s}) \leq \text{Util}(X=x; s) \quad \forall x \in \mathcal{X},$ with sharp inequality if $\text{Util}(X=x; s) > 0.$
ZEROUTIL	<i>No profits for uninformed traders:</i> $\tilde{\text{Util}}(X=x; \tilde{s}) = 0 \quad \forall x \in \mathcal{X}.$
EXUTIL	<i>Preserve excess utility:</i> $\tilde{\text{Util}}(\mu X=x; \tilde{s}) = \text{Util}(\mu X=x; s)$ for all $x \in \mathcal{X}$ and $\mu \in \mathcal{M}(X=x).$

smooth cost functions (including LMSR) they correspond to market prices that result when a trader is optimizing his guaranteed profit from the information $\omega \in \mathcal{E}$ as in Definition 2 (see Appendix E). Because of this motivation, such beliefs are referred to as “conditional price vectors.”

Definition 4. A vector $\mu \in \mathcal{M}(\mathcal{E})$ is called a conditional price vector, conditioned on \mathcal{E} , relative to the state q if $\text{Util}(\mu; q) = \text{Util}(\mathcal{E}; q)$. The set of such conditional price vectors is denoted

$$p(\mathcal{E}; q) := \{\mu \in \mathcal{M}(\mathcal{E}) : \text{Util}(\mu; q) = \text{Util}(\mathcal{E}; q)\}.$$

See Appendix F for additional motivation for our definitions of utility and conditioning. With these notions defined, we can now state our desiderata.

2.4 DESIDERATA

Recall that we aim to design mechanisms which replace a cost function C at a state s , with a new cost function \tilde{C} at a state \tilde{s} . Let Util denote the utility for information with respect to C and $\tilde{\text{Util}}$ with respect to \tilde{C} , and let p and \tilde{p} be the respective price maps. In our mechanisms, we attempt to satisfy (a subset of) the conditions on information structures as listed in Table 1.

Conditions PRICE and CONDPRICE capture the requirement to preserve the information gathered in the market. The current price $p(q)$ is the ultimate information content of the market at a state q before the observation time, but it is not necessarily the right notion of information content after the observation time. When we do not know the realization x , we may wish to set up the market so that any trader who has observed $X = x$ and would like to maximize the guaranteed profit would move the market to the same conditional price vector as in the previous market. This is captured by CONDPRICE.

DECUTIL models a scenario in which the utility for information about X decreases over time, and ZEROUTIL represents the extreme case in which utility decreases to zero. These conditions are in friction with EXUTIL, which aims

to maintain the utility structure over the conditional outcome spaces. A key challenge is to satisfy EXUTIL and ZEROUTIL (or DECUTIL) simultaneously.

Apart from the information desiderata of Table 1, we would like to maintain an important feature of cost-function-based market makers: their ability to bound the worst-case loss to the market maker. Specifically, we would like to show that there is some *finite* bound (possibly depending on the initial state) such that no matter what trades are executed and which outcome ω occurs, the market maker will lose no more than the amount of the bound. It turns out that the solution concepts introduced in this paper maintain the same loss bound as guaranteed for using just the market’s original cost function C , but since the focus of the paper is on the information structures, worst-case loss analysis is relegated to Appendix H.

In Sec. 3, we study in detail the sudden revelation setting with the goal of instantiating Protocol 1 in a way that achieves ZEROUTIL while satisfying CONDPRICE and EXUTIL. Our key result is a characterization and a geometric sufficient condition for when this is possible.

In Sec. 4, we examine instantiations of Protocol 2 for the gradual decrease setting. Our construction focuses on linearly-constrained market makers (LCMM) [13], which naturally decompose into submarkets. We show how to achieve PRICE, CONDPRICE, DECUTIL and EXUTIL in LCMMs. We also show that it is possible to simultaneously decrease the utility for information in each submarket according to its own schedule, while maintaining PRICE.

Before we develop these mechanisms, we introduce the machinery of Bregman divergences, which helps us analyze notions of utility for information.

2.5 BREGMAN DIVERGENCE AND UTILITY

To analyze the market maker’s utility for information, we show how it corresponds to a specific notion of distance built into the cost function, the *mixed (or generalized) Bregman divergence* [13, 15]. Let R be the conjugate of C .³ The mixed Bregman divergence between a belief μ and a state q is defined as $D(\mu||q) := R(\mu) + C(q) - q \cdot \mu$. The conjugacy of R and C implies that $D(\mu||q) \geq 0$ with equality iff $\mu \in \partial C(q) = p(q)$, i.e., if the price vector “matches” the state (see Appendix A). The geometric interpretation of mixed Bregman divergence is as a gap between a tangent and the graph of the function R (see Fig. 1).

To see how the divergence relates to traders’ beliefs, consider a trader who believes that $\mathbb{E}[\rho(\omega)] = \mu'$ and moves the market from state q to state q' . The expected payoff to this trader is $(q' - q) \cdot \mu' - C(q') + C(q) = D(\mu'||q) - D(\mu'||q')$. This payoff increases as $D(\mu'||q')$

³The conjugate is also, less commonly, called the “dual”.

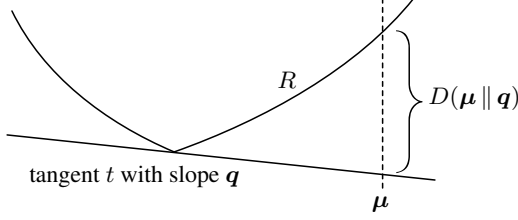


Figure 1: The mixed Bregman divergence $D(\mu||q)$ derived from the conjugate pair C and R measures the distance between the tangent with slope q and the value of R evaluated at μ . By conjugacy, the tangent t is described by $t(\mu) = \mu \cdot q - C(q)$. Note that the divergence is well defined even when R is not differentiable, because each slope vector determines a unique tangent.

decreases. Thus, subject to the trader’s budget constraints, the trader is incentivized to move to the state q' which is as “close” to his/her belief μ' as possible in the sense of a smaller value $D(\mu'||q')$, with the largest expected payoff when $D(\mu'||q') = 0$. This argument shows that $D(\cdot||\cdot)$ is an implicit measure of distance used by traders.

The next theorem shows that the Bregman divergence also matches the concepts defined in Sec. 2.3. Specifically, we show that (1) the utility for a belief coincides with the Bregman divergence, (2) the utility for an event \mathcal{E} is the smallest divergence between the current market state and $\mathcal{M}(\mathcal{E})$, and (3) the conditional price vector is the (Bregman) projection of the current market state on $\mathcal{M}(\mathcal{E})$, i.e., it is a belief in $\mathcal{M}(\mathcal{E})$ that is “closest to” the current market state.

Theorem 1. *Let $\mu \in \mathcal{M}$, $q \in \mathbb{R}^K$ and $\emptyset \neq \mathcal{E} \subseteq \Omega$. Then*

$$\text{Util}(\mu; q) = D(\mu||q) , \quad (1)$$

$$\text{Util}(\mathcal{E}; q) = \min_{\mu' \in \mathcal{M}(\mathcal{E})} D(\mu'||q) , \quad (2)$$

$$p(\mathcal{E}; q) = \operatorname{argmin}_{\mu' \in \mathcal{M}(\mathcal{E})} D(\mu'||q) . \quad (3)$$

We finish this section by characterizing when EXUTIL is satisfied and showing that it implies CONDPRICE. Recall that $\Omega^x = \{\omega : X(\omega) = x\}$ and let $\mathcal{M}^x := \mathcal{M}(\Omega^x)$.

Proposition 1. *EXUTIL holds if and only if for all $x \in X$, there exists some c^x such that for all $\mu \in \mathcal{M}^x$, $D(\mu||s) - \tilde{D}(\mu||\tilde{s}) = c^x$. Moreover, EXUTIL implies CONDPRICE.*

3 SUDDEN REVELATION

In this section, we consider the design of sudden revelation market makers (Protocol 1). In this setting, partial information in the form of the realization of X is revealed to market participants (but not to the market maker) at a predetermined time, as might be the case if the medal winners of an Olympic event are announced but no human is available to input this information into the automated market maker on behalf of the market organizer. The random variable X and the observation time are assumed to be known, and the market maker wishes to “close” the submarket with respect to X just before the observation time, without knowing the realization x , while leaving the rest of the market unchanged.

Stated in terms of our formalism, we wish to find functions `NewState` and `NewCost` from Protocol 1 such that the desiderata CONDPRICE, EXUTIL, and ZEROUTIL from Table 1 are satisfied. This implies that traders who know only that $X = x$ are not rewarded after the observation time, but traders with new information about the outcome space conditioned on $X = x$ are rewarded exactly as before. As a result, trading immediately resumes in a “conditional market” on $\mathcal{M}(\Omega^x)$ for the correct realization x , without the market maker needing to know x and without any other human intervention. We refer to the goal of simultaneously achieving CONDPRICE, EXUTIL, and ZEROUTIL as achieving *implicit submarket closing*.

For convenience, throughout this section we write $\mathcal{M}^x := \mathcal{M}(\Omega^x)$ to denote the conditional price space, and $\mathcal{M}^* := \bigcup_{x \in \mathcal{X}} \mathcal{M}^x$ to denote prices possible after the observation.

3.1 SIMPLIFYING THE OBJECTIVE

We first show that achieving implicit submarket closing can be reduced to finding a function \tilde{R} satisfying a simple set of constraints, and defining `NewCost` to return the conjugate \tilde{C} of \tilde{R} . As a first step, we observe that it is without loss of generality to let `NewState` be an identity map, i.e., to assume that $\tilde{s} = s$; when this is not the case, we can obtain an equivalent market by setting $\tilde{s} = s$ and shifting \tilde{C} so that the Bregman divergence is unchanged.

Lemma 1. *Any desideratum of Table 1 holds for \tilde{C} and \tilde{s} if and only if it holds for $\tilde{C}'(q) = \tilde{C}(q + \tilde{s} - s)$ and $\tilde{s}' = s$.*

To simplify exposition, we assume that $\tilde{s} = s$ throughout the rest of the section as we search for conditions on `NewCost` that achieve implicit submarket closing. Under this assumption, Proposition 1 can be used to characterize our goal in terms of \tilde{R} . Specifically, we show that EXUTIL and CONDPRICE hold if \tilde{R} differs from R by a (possibly different) constant on each conditional price space \mathcal{M}^x .

Lemma 2. *When $\tilde{s} = s$, EXUTIL and CONDPRICE hold together if and only if there exist constants b^x for $x \in \mathcal{X}$ such that $\tilde{R}(\mu) = R(\mu) - b^x$ for all $x \in \mathcal{X}$ and $\mu \in \mathcal{M}^x$.*

This suggests parameterizing our search for \tilde{R} by vectors $b = \{b^x\}_{x \in \mathcal{X}}$. For $b \in \mathbb{R}^{\mathcal{X}}$, define a function

$$R^b(\mu) = \begin{cases} R(\mu) - b^x & \text{if } \mu \in \mathcal{M}^x, x \in \mathcal{X}, \\ \infty & \text{otherwise.} \end{cases}$$

If the sets \mathcal{M}^x overlap, R^b is not well defined for all b . Whenever we write R^b , we assume that b is such that R^b is well defined. To satisfy Lemma 2 with a specific b , it suffices to find a convex function \tilde{R} “consistent with” R^b in the following sense.

Definition 5. *We say that a function \tilde{R} is consistent with R^b if $\tilde{R}(\mu) = R^b(\mu)$ for all $\mu \in \mathcal{M}^*$.*

We next simplify our objective further by proving that

whenever implicit submarket closing is achievable, it suffices to consider functions NewCost that set \tilde{C} to be the conjugate of the largest convex function consistent with R^b for some $b \in \mathbb{R}^{\mathcal{X}}$. To establish this, we examine properties of the *convex roof* of R^b , the largest convex function that lower-bounds (but is not necessarily consistent with) R^b .

Definition 6. Given a function $f : \mathbb{R}^K \rightarrow (-\infty, \infty]$, the convex roof of f , denoted $(\text{conv } f)$, is the largest convex function lower-bounding f , defined by

$$(\text{conv } f)(x) := \sup \{g(x) : g \in \mathcal{G}, g \leq f\}$$

where \mathcal{G} is the set of convex functions $g : \mathbb{R}^K \rightarrow (-\infty, \infty]$, and the condition $g \leq f$ holds pointwise.

The convex roof is analogous to a convex hull, and the epigraph of $(\text{conv } f)$ is the convex hull of the epigraph of f . See Urruty and Lemarchal [30, §B.2.5] for details.

Example 4. Recall the square market of Example 2. Let $X(\omega) = \omega_1$, so traders observe the payoff of the first security at observation time. Then $\mathcal{M}^x = \{x\} \times [0, 1]$ for $x \in \{0, 1\}$. For simplicity, let $b = 0$. We have $R^b(\mu) = \mu_2 \ln \mu_2 + (1 - \mu_2) \ln(1 - \mu_2)$ for $\mu \in \mathcal{M}^1 \cup \mathcal{M}^2$ and $R^b(\mu) = \infty$ for all other μ . Examining the convex hull of the epigraph of R^b gives us that for all $\mu \in [0, 1]^2$, we have $(\text{conv } R^b)(\mu) = \mu_2 \ln \mu_2 + (1 - \mu_2) \ln(1 - \mu_2)$.

As this example illustrates, the roof of R^b is the “flattest” convex function lower-bounding R^b . Given the geometric interpretation of Bregman divergence (Fig. 1), a “flatter” \tilde{R} yields a smaller utility for information. This flatness plays a key role in achieving ZEROUTIL. Assume that \tilde{R} is consistent with R^b , so CONDPRICE and EXUTIL hold by Lemma 2. Following the intuition in Fig. 1, to achieve ZEROUTIL, i.e., $\tilde{D}(\hat{\mu}^x \| s) = 0$ across all $x \in \mathcal{X}$ and $\hat{\mu}^x \in p(\Omega^x; s)$, it must be the case that for all x and $\hat{\mu}^x$, the function values $\tilde{R}(\hat{\mu}^x)$ lie on the tangent of \tilde{R} with slope s . That is, the graph of \tilde{R} needs to be *flat* across the points $\hat{\mu}^x$. This suggests that the roof might be a good candidate for \tilde{R} . This intuition is formalized in the following lemma, which states that instead of considering arbitrary convex \tilde{R} consistent with R^b , we can consider \tilde{R} which take the form of a convex roof.

Lemma 3. If any convex function \tilde{R} is consistent with R^b then so is the convex roof $\tilde{R}' = (\text{conv } R^b)$. Furthermore, if \tilde{R} satisfies ZEROUTIL or DECUTIL then so does \tilde{R}' .

3.2 IMPLICIT SUBMARKET CLOSING

We now have the tools to answer the central question of this section: When can we achieve implicit submarket closing? Lemma 1 implies that we can assume that NewState is the identity function, and Lemmas 2 and 3 imply that it suffices to consider functions NewCost that set \tilde{C} to the conjugate of $\tilde{R} = (\text{conv } R^b)$ for some $b \in \mathbb{R}^{\mathcal{X}}$. What remains is to find the vector b that guarantees ZEROUTIL. As mentioned above, ZEROUTIL is satisfied if and only if $(\hat{\mu}^x, \tilde{R}(\hat{\mu}^x))$

lies on the tangent of \tilde{R} with the slope s for all $x \in \mathcal{X}$ and $\hat{\mu}^x \in p(\Omega^x; s)$. This implies that $\tilde{R}(\hat{\mu}^x) = \hat{\mu}^x \cdot s - c$ for all x and $\hat{\mu}^x$ and some constant c . The specific choice of c does not matter since \tilde{D} is unchanged by vertical shifts of the graph of \tilde{R} . For convenience, we set $c = C(s)$, which makes the tangents of R and \tilde{R} with the slope s coincide. This and Lemma 2 then yield the choice of $b = \hat{b}$, with

$$\hat{b}^x := R(\hat{\mu}^x) + C(s) - \hat{\mu}^x \cdot s = D(\hat{\mu}^x \| s) \quad (4)$$

for all x and any choice of $\hat{\mu}^x \in p(\Omega^x; s)$. The resulting construction of $\tilde{R} = (\text{conv } R^{\hat{b}})$ can be described using geometric intuition. First, consider the tangent of R with slope equal to the current market state s . For each $x \in \mathcal{X}$, take the subgraph of R over the set \mathcal{M}^x and let it “fall” vertically until it touches this tangent at the point $\hat{\mu}^x$. The set of fallen graphs for all x together describes $R^{\hat{b}}$ and the convex hull of the fallen epigraphs yields $\tilde{R} = (\text{conv } R^{\hat{b}})$.

Defining NewCost using this construction guarantees ZEROUTIL, but CONDPRICE and EXUTIL are achieved only when \tilde{R} is consistent with $R^{\hat{b}}$. Conversely, whenever the three properties are achievable, this construction produces a function \tilde{R} consistent with $R^{\hat{b}}$. This yields a full characterization of when implicit submarket closing is achievable.

Theorem 2. Let \hat{b} be defined as in Eq. (4). CONDPRICE, EXUTIL, and ZEROUTIL can be satisfied using Protocol 1 if and only if $(\text{conv } R^{\hat{b}})$ is consistent with $R^{\hat{b}}$. In this case, they can be achieved with NewState as the identity and NewCost outputting the conjugate of $\tilde{R} = (\text{conv } R^{\hat{b}})$.

3.3 CONSTRUCTING THE COST FUNCTION

Theorem 2 describes how to achieve implicit submarket closing by defining the cost function \tilde{C} output by NewCost implicitly via its conjugate \tilde{R} . In this section, we provide an explicit construction of the resulting cost function, and illustrate the construction through examples.

Fixing R , for each $x \in \mathcal{X}$ define a function $C^x(q) := \sup_{\mu \in \mathcal{M}^x} [q \cdot \mu - R(\mu)]$. Each function C^x can be viewed as a bounded-loss and arbitrage-free cost function for outcomes in Ω^x . The conjugate of each C^x coincides with R on \mathcal{M}^x (and is infinite outside \mathcal{M}^x). The explicit expression for \tilde{C} is described in the following proposition.

Proposition 2. For a given C with conjugate R , define \hat{b} as in Eq. (4) and let $\tilde{R} = (\text{conv } R^{\hat{b}})$. The conjugate \tilde{C} of \tilde{R} can be written $\tilde{C}(q) = \max_{x \in \mathcal{X}} [\hat{b}^x + C^x(q)]$. Furthermore, for each $x \in \mathcal{X}$, $\hat{b}^x = C(s) - C^x(s)$.

At any market state q with a unique $\hat{x} := \arg\max_{x \in \mathcal{X}} [\hat{b}^x + C^x(q)]$, the price according to \tilde{C} lies in the set $\mathcal{M}^{\hat{x}}$. When \hat{x} is not unique, the market has a bid-ask spread. The addition of \hat{b}^x ensures that the bid-ask spread at the market state s contains conditional prices $\hat{\mu}^x$ across all x . To illustrate this construction, we return to the example of a square.

Example 5. Consider again the square market from Examples 2 and 4 with $X(\omega) = \omega_1$. One can verify that $C^x(q) = xq_1 + \ln(1 + e^{q_2})$ for $x \in \{0, 1\}$. Prop. 2 gives

$$\begin{aligned}\tilde{C}(q) &= \max_{x \in \{0, 1\}} [x(q_1 - s_1) + \ln(1 + e^{q_2}) + \ln(1 + e^{s_1})] \\ &= \max\{0, q_1 - s_1\} + \ln(1 + e^{s_1}) + \ln(1 + e^{q_2}).\end{aligned}$$

In switching from C to \tilde{C} we have effectively changed the first term of our cost from a basic LMSR cost for a single binary security to the piecewise linear cost of Example 3, introducing a bid-ask spread for security 1 when $q_1 = s_1$; states $q = (s_1, q_2)$ have $\tilde{p}(q) = [0, 1] \times \{e^{q_2}/(1 + e^{q_2})\}$. The market for security 1 has thus implicitly closed; as the new market begins with $q = s$, any trader can switch the price of security 1 to 0 or 1 by simply purchasing an infinitesimal quantity of security 1 in the appropriate direction, at essentially no cost and with no ability to profit.

The example above illustrates our cost function construction, but does not show that \tilde{R} is consistent with R^b as required by Theorem 2. In fact, it is consistent. This follows from the sufficient condition proved in Appendix G.2. Briefly, the condition is that \mathcal{M}^* does not contain any price vectors μ that can be expressed as nontrivial convex combinations of vectors from multiple \mathcal{M}^x .

In Appendix G.3, we show that this sufficient condition applies to many settings of interest such as arbitrary partitions of simplex and submarket observations in binary-payoff LCMMs (defined in Sec. 4), which were used to run a combinatorial market for the 2012 U.S. Elections [14].

A case in which the sufficient condition is violated is the square market with $X(\omega) = \omega_1 + \omega_2 \in \{0, 1, 2\}$, where $\mathcal{M}^0 = (0, 0)$ and $\mathcal{M}^2 = (1, 1)$ but $(\frac{1}{2}, \frac{1}{2}) = \frac{1}{2}(0, 0) + \frac{1}{2}(1, 1) \in \mathcal{M}^1$. This particular example also fails to satisfy Theorem 2 (see Appendix G.1), but in general the sufficient condition is not necessary (see Appendix G.4).

4 GRADUAL DECREASE

We now consider gradual decrease market makers (Protocol 2) for the gradual decrease setting in which the utility of information about a future observation X is decreasing continuously over time. We focus on *linearly constrained market makers* (LCMMs) [13], which naturally decompose into submarkets. Our proposed gradual decrease market maker employs a different LCMM at each time step, and satisfies various desiderata of Sec. 2.4 between steps.

As a warm-up for the concepts introduced in this section, we show how the “liquidity parameter” can be used to implement a decreasing utility for information.

Example 6. *Homogeneous decrease in utility for information.* We begin with a differentiable cost function C in a state s . Let $\alpha \in (0, 1)$, and define $\tilde{C}(q) = \alpha C(q/\alpha)$, and $\tilde{s} = \alpha s$. \tilde{C} is parameterized by the “liquidity parameter” α .

The transformation \tilde{s} guarantees the preservation of prices, i.e., $\tilde{p}(\tilde{s}) = \nabla \tilde{C}(\tilde{s}) = \alpha \nabla C(\tilde{s}/\alpha)/\alpha = \nabla C(s) = p(s)$. We can derive that $\tilde{R}(\mu) = \alpha R(\mu)$, and $\tilde{D}(\mu||q) = \alpha D(\mu||q/\alpha)$, so, for all μ , $\tilde{D}(\mu||\tilde{s}) = \alpha D(\mu||s)$. In words, the utility for all beliefs μ with respect to the current state is decreased according to the multiplier α .

This idea will be the basis of our construction. We next define the components of our setup and prove the desiderata.

4.1 LINEARLY CONSTRAINED MARKETS

Recall that $\rho : \Omega \rightarrow \mathbb{R}^K$ is the payoff function. Let \mathcal{G} be a system of non-empty disjoint subsets $g \subseteq [K]$ forming a partition of coordinates of ρ , so $[K] = \bigcup_{g \in \mathcal{G}} g$. We use the notation $\rho_g(\omega) := (\rho_i(\omega))_{i \in g}$ for the block of coordinates in g , and similarly μ_g and q_g . Blocks g describe groups of securities that are treated as separate “submarkets,” but there can be logical dependencies among them.

Example 7. *Medal counts.* Consider a prediction market for the Olympics. Assume that Norway takes part in n Olympic events. In each, Norway can win a gold medal or not. Encode this outcome space as $\Omega = \{0, 1\}^n$. Define random variables $X_i(\omega) = \omega_i$ equal to 1 iff Norway wins gold in the i th Olympic event. Also define a random variable $Y = \sum_{i=1}^n X_i$ representing the number of gold medals that Norway wins in total. We create $K = 2n + 1$ securities, corresponding to 0/1 indicators of the form $1[X_i = 1]$ for $i \in [n]$ and $1[Y = y]$ for $y \in \{0, \dots, n\}$. That is, $\rho_i = X_i$ for $i \in [n]$ and $\rho_{n+1+y} = 1[Y = y]$ for $y \in \{0, \dots, n\}$. A natural block structure in this market is $\mathcal{G} = \{\{1\}, \{2\}, \dots, \{n\}, \{n+1, \dots, 2n+1\}\}$ with submarkets corresponding to the X_i and Y .

Given the block structure \mathcal{G} , the construction of a linearly constrained market begins with bounded-loss and arbitrage-free convex cost functions $C_g : \mathbb{R}^g \rightarrow \mathbb{R}$ with conjugates R_g and divergences D_g for each $g \in \mathcal{G}$. These cost functions are assumed to be easy to compute and give rise to a “direct-sum” cost $C_\oplus(q) = \sum_{g \in \mathcal{G}} C_g(q_g)$ with the conjugate $R_\oplus(\mu) = \sum_{g \in \mathcal{G}} R_g(\mu_g)$ and divergence $D_\oplus(\mu||q) = \sum_{g \in \mathcal{G}} D_g(\mu_g||q_g)$.

Since C_\oplus decomposes, it can be calculated quickly. However, the market maker C_\oplus might allow arbitrage due to the lack of consistency among submarkets since arbitrage opportunities arise when prices fall outside \mathcal{M} [1]. \mathcal{M} is always polyhedral, so it can be described as $\mathcal{M} = \{\mu \in \mathbb{R}^K : \mathbf{A}^\top \mu \geq \mathbf{b}\}$ for some matrix $\mathbf{A} \in \mathbb{R}^{K \times M}$ and vector $\mathbf{b} \in \mathbb{R}^M$. Letting \mathbf{a}_m denote the m th column of \mathbf{A} , arbitrage opportunities open up if the price of the bundle \mathbf{a}_m falls below b_m . For any $\eta \in \mathbb{R}_+^M$, the bundle $\mathbf{A}\eta$ presents an arbitrage opportunity if priced below $\mathbf{b} \cdot \eta$.

A *linearly constrained market maker* (LCMM) is described by the cost function $C(q) = \inf_{\eta \in \mathbb{R}_+^M} [C_\oplus(q + \mathbf{A}\eta) - \mathbf{b} \cdot \eta]$. While the definition of C is slightly involved, the conju-

gate R has a natural meaning as a restriction of the direct-sum market to the price space \mathcal{M} , i.e., $R(\mu) = R_{\oplus}(\mu) + \mathbb{I}[\mu \in \mathcal{M}]$. Furthermore, the infimum in the definition of C is always attained (see Appendix D.1). Fixing q and letting η^* be a minimizer in the definition, we can think of the market maker as automatically charging traders for the bundle $A\eta^*$, which would present an arbitrage opportunity, and returning to them the guaranteed payout $b \cdot \eta$. This benefits traders while maintaining the same worst-case loss guarantee for the market maker as C_{\oplus} [13].

Example 8. LCMM for medal counts. Continuing the previous example, for submarkets X_i , we can define LMSR costs $C_i(q_i) = \ln(1 + \exp(q_i))$. For the submarket for Y , let $g = \{n+1, \dots, 2n+1\}$ and use the LMSR cost $C_g(q_g) = \ln(\sum_{y=0}^n \exp(q_{n+1+y}))$. The submarkets for X_i and Y are linked. One example of a linear constraint is based on the linearity of expectations: for any distribution, we must have $\mathbb{E}[Y] = \sum_{i=1}^n \mathbb{E}[X_i]$. This places an equality constraint $\sum_{y=0}^n y \cdot \mu_{n+1+y} = \sum_{i=1}^n \mu_i$ on the vector μ , which can be expressed as two inequality constraints (see Dudík et al. [13, 14] for more on constraint generation).

4.2 DECREASING LIQUIDITY

We now study the gradual decrease scenario in which the utility for information in each submarket g decreases over time. In the Olympics example, the market maker may want to continuously decrease the rewards for information about a particular event as the event takes place.

We generalize the strategy from Example 6 to LCMMs and extend them to time-sensitive cost functions by introducing the “information-utility schedule” in the form of a differentiable non-increasing function $\beta_g : \mathbb{R} \rightarrow (0, 1]$ with $\beta_g(t^0) = 1$. The speed of decrease of β_g controls the speed of decrease of the utility for information in each submarket. (We make this statement more precise in Theorem 3.)

We first define a gradual decrease direct-sum cost function $C_{\oplus}(q; t) = \sum_{g \in \mathcal{G}} \beta_g(t) C_g(q_g / \beta_g(t))$ which is used to define a gradual decrease LCMM, and a matching NewState as follows:

$$\begin{aligned} C(q; t) &= \inf_{\eta \in \mathbb{R}_+^M} [C_{\oplus}(q + A\eta; t) - b \cdot \eta] \\ \text{NewState}(q; t, \tilde{t}) &= \tilde{q} \\ \text{such that } \tilde{q}_g &= \frac{\beta_g(\tilde{t})}{\beta_g(t)} (q_g + \delta_g^*) - \delta_g^* \\ \text{where } \eta^* &\text{ is a minimizer in } C(q; t) \text{ and } \delta^* = A\eta^* . \end{aligned}$$

When considering the state update from time t to time \tilde{t} , the ratio $\beta_g(\tilde{t})/\beta_g(t)$ has the role of the liquidity parameter α in Example 6. The motivation behind the definition of NewState is to guarantee that $\tilde{q}_g + \delta_g^* = [\beta_g(\tilde{t})/\beta_g(t)](q_g + \delta_g^*)$, which turns out to ensure that η^* remains the minimizer and the prices are unchanged. The preservation of prices (PRICE) is achieved by a scaling sim-

ilar to Example 6, albeit applied to the market state in the direct-sum market underlying the LCMM.

This intuition is formalized in the next theorem, which shows that the above construction preserves prices and decreases the utility for information, as captured by the mixed Bregman divergence, according to the schedules β_g . We use the notation $C^t(q) := C(q; t)$ and write D_g^t for the divergence derived from $C_g^t(q_g) := \beta_g(t) C_g(q_g / \beta_g(t))$.

Theorem 3. *Let C be a gradual decrease LCMM, let $t, \tilde{t} \in \mathbb{R}$ and $s \in \mathbb{R}^K$. The replacement of C^t by $\tilde{C} := C^{\tilde{t}}$ and s by $\tilde{s} := \text{NewState}(s; t, \tilde{t})$ satisfies PRICE. Also,*

$$\tilde{D}(\mu \| \tilde{s}) = \sum_{g \in \mathcal{G}} \alpha_g D_g^t(\mu_g \| s_g + \delta_g^*) + (A^\top \mu - b) \cdot \eta^* \quad (5)$$

for all $\mu \in \mathcal{M}$, where η^* and δ^* are defined by $\text{NewState}(s; t, \tilde{t})$, and $\alpha_g = \beta_g(\tilde{t})/\beta_g(t) > 0$.

The first term on the right-hand side of Eq. (5) is the sum of divergences in submarkets g , each weighted by a coefficient α_g which is equal to one at $\tilde{t} = t$ and weakly decreases as \tilde{t} grows. The divergences are between μ_g and the state resulting from the arbitrage action in the direct-sum market. The second term is non-negative, since $\mu \in \mathcal{M}$, and represents expected arbitrage gains beyond the guaranteed profit from the arbitrage in the direct-sum market. The only terms that depend on time \tilde{t} are the multipliers α_g . Since they are decreasing over time, we immediately obtain that the utility for information, $\text{Util}(\mu; \tilde{s}) = \tilde{D}(\mu \| \tilde{s})$, is also decreasing, with the contributions from individual submarkets decreasing according to their schedules β_g .

When only one of the schedules β_g is decreasing and the other schedules stay constant, we can show that the excess utility and conditional prices are preserved (conditioned on ρ_g), and under certain conditions also DECUTIL holds.

For a submarket g , let $\mathcal{X}_g := \{\rho_g(\omega) : \omega \in \Omega\}$ be the set of realizations of ρ_g . Recall that $\mathcal{M}(\mathcal{E})$ is the convex hull of $\{\rho(\omega)\}_{\omega \in \mathcal{E}}$. We show that DECUTIL holds if C_g is differentiable and the submarket g is “tight” as follows.

Definition 7. *We say that a submarket g is tight if for all $x \in \mathcal{X}_g$ the set $\{\mu \in \mathcal{M} : \mu_g = x\}$ coincides with $\mathcal{M}(\rho_g = x)$, i.e., if all the beliefs μ with $\mu_g = x$ can be realized by probability distributions over states ω with $\rho_g(\omega) = x$. (In general, the former is always a superset of the latter, hence the name “tight” when the equality holds.)*

While this condition is somewhat restrictive, it is easy to see that all submarkets with binary securities, i.e., with $\rho_g(\omega) \in \{0, 1\}^g$, are tight (see Appendix D.4).

Theorem 4. *Assume the setup of Theorem 3. Let $g \in \mathcal{G}$ and assume that $\beta_g(\tilde{t}) < \beta_g(t)$ whereas $\beta_{g'}(\tilde{t}) = \beta_{g'}(t)$ for $g' \neq g$. Then the replacement of C^t by \tilde{C} and s by \tilde{s} satisfies CONDPRICE and EXUTIL for the random variable ρ_g . Furthermore, if C_g is differentiable and the submarket g is tight, we also obtain DECUTIL.*

References

- [1] Jacob Abernethy, Yiling Chen, and Jennifer Wortman Vaughan. An optimization-based framework for automated market-making. In *Proceedings of the 12th ACM Conference on Electronic Commerce*, 2011.
- [2] Jacob Abernethy, Yiling Chen, and Jennifer Wortman Vaughan. Efficient market making via convex optimization, and a connection to online learning. *ACM Transactions on Economics and Computation*, 1(2):12:1–12:39, 2013.
- [3] Jacob Abernethy, Rafael Frongillo, Xiaolong Li, and Jennifer Wortman Vaughan. A general volume-parameterized market making framework. In *Proceedings of the 15th ACM Conference on Economics and Computation*, 2014.
- [4] Joyce Berg, Robert Forsythe, Forrest Nelson, and Thomas Rietz. Results from a dozen years of election futures markets research. In Charles R. Plott and Vernon L. Smith, editors, *Handbook of Experimental Economics Results*, volume 1, pages 742–751. Elsevier, 2008.
- [5] Aseem Bhatta, Mithun Chakraborty, Sanmay Das, Allen Lavoie, and Malik Magdon-Ismael. A Bayesian market maker. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, 2012.
- [6] Philip Delves Broughton. Prediction markets: Value among the crowd. *Financial Times*, April 2013.
- [7] Robert Charette. An internal futures market. *Information Management*, March 2007.
- [8] Y. Chen, L. Fortnow, E. Nikolova, and D.M. Pennock. Betting on permutations. In *Proceedings of the 8th ACM Conference on Electronic Commerce*, 2007.
- [9] Y. Chen, L. Fortnow, N. Lambert, D. M. Pennock, and J. Wortman. Complexity of combinatorial market makers. In *Proceedings of the 9th ACM Conference on Electronic Commerce*, 2008.
- [10] Y. Chen, S. Goel, and D.M. Pennock. Pricing combinatorial markets for tournaments. In *ACM Symposium on Theory of Computing*, 2008.
- [11] Yiling Chen and David M. Pennock. A utility framework for bounded-loss market makers. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence*, 2007.
- [12] Bo Cowgill, Justin Wolfers, and Eric Zitzewitz. Using prediction markets to track information flows: Evidence from Google. Working paper, 2008.
- [13] Miroslav Dudík, Sébastien Lahaie, and David M. Pennock. A tractable combinatorial market maker using constraint generation. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, 2012.
- [14] Miroslav Dudík, Sébastien Lahaie, David M. Pennock, and David Rothschild. A combinatorial prediction market for the U.S. elections. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, 2013.
- [15] Geoffrey J. Gordon. Regret bounds for prediction problems. In *Proceedings of the 12th Annual Conference on Computational Learning Theory*, pages 29–40, 1999.
- [16] P.D. Grünwald and A.P. Dawid. Game theory, maximum entropy, minimum discrepancy and robust Bayesian decision theory. *The Annals of Statistics*, 32(4):1367–1433, 2004.
- [17] M. Guo and D.M. Pennock. Combinatorial prediction markets for event hierarchies. In *International Conference on Autonomous Agents and Multiagent Systems*, 2009.
- [18] R. Hanson. Combinatorial information market design. *Information Systems Frontiers*, 5(1):105–119, 2003.
- [19] R. Hanson. Logarithmic market scoring rules for modular combinatorial information aggregation. *Journal of Prediction Markets*, 1(1):3–15, 2007.
- [20] Jean-Baptiste Hiriart-Urruty and Claude Lemaréchal. *Convex Analysis and Minimization Algorithms*, volume 1. Springer, 1996.
- [21] Xiaolong Li and Jennifer Wortman Vaughan. An axiomatic characterization of adaptive-liquidity market makers. In *Proceedings of the 14th ACM Conference on Electronic Commerce*, 2013.
- [22] A. Othman and T. Sandholm. Liquidity-sensitive automated market makers via homogeneous risk measures. In *Proceedings of the 7th Workshop on Internet and Network Economics*, 2011.
- [23] A. Othman and T. Sandholm. Profit-charging market makers with bounded loss, vanishing bid/ask spreads, and unlimited market depth. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, 2012.
- [24] A. Othman, T. Sandholm, D.M. Pennock, and D.M. Reeves. A practical liquidity-sensitive automated market maker. In *Proceedings of the 11th ACM Conference on Electronic Commerce*, 2010.
- [25] David M. Pennock, Steve Lawrence, C. Lee Giles, and Finn A. Nielsen. The real power of artificial markets. *Science*, 291:987–988, 2002.
- [26] D.M. Pennock and L. Xia. Price updating in combinatorial prediction markets with Bayesian networks. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence*, 2011.
- [27] R. Tyrrell Rockafellar. *Convex Analysis*. Princeton University Press, 1970.
- [28] Martin Spann and Bernd Skiera. Internet-based virtual stock markets for business forecasting. *Management Science*, 49(10):1310–1326, 2003.
- [29] Lyle Ungar, Barb Mellors, Ville Satopää, Jon Baron, Phil Tetlock, Jaime Ramos, and Sam Swift. The good judgment project: A large scale test of different methods of combining expert predictions. AAAI Technical Report FS-12-06, 2012.
- [30] Jean-Baptiste Hiriart Urruty and Claude Lemarchal. *Fundamentals of Convex Analysis*. Springer, 2001.

Universal Convexification via Risk-Aversion

Krishnamurthy Dvijotham

Dept of Computer Science & Engg
University of Washington
Seattle, WA 98195

Maryam Fazel

Dept of Electrical Engg
University of Washington
Seattle, WA 98195

Emanuel Todorov

Dept of Computer Science & Engg
Dept of Applied Mathematics
University of Washington
Seattle, WA 98195

Abstract

We develop a framework for convexifying a general class of optimization problems. We analyze the suboptimality of the solution to the convexified problem relative to the original nonconvex problem, and prove additive approximation guarantees under some assumptions. In simple settings, the convexification procedure can be applied directly and standard optimization methods can be used. In the general case we rely on stochastic gradient algorithms, whose convergence rate can be bounded using the convexity of the underlying optimization problem. We then extend the framework to a general class of discrete-time dynamical systems where our convexification approach falls under the paradigm of risk-sensitive Markov Decision Processes. We derive the first model-based and model-free policy gradient optimization algorithms with guaranteed convergence to the optimal solution. We also present numerical results in different machine learning applications.

1 INTRODUCTION

It has been said that the watershed in optimization is not between linearity and nonlinearity, but between convexity and nonconvexity. In this paper we develop a framework for convexifying a general class of optimization problems (section 3), turning them into problems that can be solved with efficient convergence guarantees. The convexification approach may change the problem drastically in some cases, which is not surprising since most nonconvex optimization problems are NP-hard and cannot be reduced to solving convex optimization problems. However, under additional assumptions, we obtain guarantees that bound the suboptimality of the solution relative to the original non-

convex problem (section 3.2). We adapt stochastic gradient methods to solve the resulting problems (section 4) and prove convergence guarantees that bound the distance to optimality as a function of the number of iterations. In section 5, we extend the framework to arbitrary dynamical systems and derive the first policy optimization approach with guaranteed convergence to the global optimum. The control problems we study fall under the framework of risk-sensitive control. The condition required for convexity is a natural one, relating the control cost, risk factor and noise covariance. It is very similar to the condition that reduces stochastic optimal control to a linear problem [Fleming and Mitter, 1982, Kappen, 2005, Todorov, 2009], which in turn has given rise to path-integral and other specialized methods for control [Theodorou et al., 2010a, Broek et al., 2010, Dvijotham and Todorov, 2011, Toussaint, 2009].

Smoothing with noise is a relatively common approach to simplify difficult optimization problems. It has been used in computer vision [Rangarajan, 1990] heuristically and formalized in recent work [Mobahi, 2012] where it is shown that under certain assumptions, adding sufficient noise eventually leads to a convex optimization problem. In contrast, we show that for any noise level one can choose the degree of risk-aversion so as to obtain a convex problem. This procedure may destroy minima that are not robust to perturbations – which is not necessarily undesirable, because in many applications it makes sense to seek a robust solution rather than the best one.

2 NOTATION

Gaussian random variables are denoted by $\omega \sim \mathcal{N}(\mu, \Sigma)$, where μ is the mean and Σ the covariance matrix. Given a random variable $\omega \in \Omega$ with distribution P and a function $h : \Omega \mapsto \mathbf{R}$, the expected value of the random variable $h(\omega)$ is written as $\mathbb{E}_{\omega \sim P}[h(\omega)]$. Whenever it is clear from the context, we will drop

the subscript on the expected value and simply write $E[h(\omega)]$. We differ from convention here slightly by denoting random variables with lowercase letters and reserve uppercase letters for matrices. We will frequently work with Gaussian perturbations of a function, so we denote:

$$\bar{h}(\theta) = E_{\omega \sim \mathcal{N}(0, \Sigma)} [h(\theta + \omega)], \tilde{h}_\omega(\theta) = h(\theta + \omega) - \bar{h}(\theta).$$

I_n denotes the $n \times n$ identity matrix. Unless stated otherwise, $\|\cdot\|$ refers to the Euclidean ℓ_2 norm. Given a convex set \mathcal{C} , we denote the projection onto \mathcal{C} of x by $\text{Proj}_{\mathcal{C}}(x)$: $\text{Proj}_{\mathcal{C}}(x) = \text{argmin}_{y \in \mathcal{C}} \|y - x\|$. The maximum eigenvalue of a symmetric matrix M is denoted by $\lambda_{\max}(M)$. Given symmetric matrices A, B , the notation $A \succeq B$ means that $A - B$ is a positive semidefinite matrix. Given a positive definite matrix $M \succ 0$, we denote the metric induced by M as $\|x\|_M = \sqrt{x^T M x}$.

3 GENERAL OPTIMIZATION PROBLEMS

We study optimization problems of the form:

$$\min_{\theta \in \mathcal{C}} g(\theta) \quad (1)$$

where g is an arbitrary function and $\mathcal{C} \subset \mathbf{R}^k$ is a convex set. We do not assume that g is convex so the above problem could be a nonconvex optimization problem. In this work, we convexify this problem by decomposing $g(\theta)$ as follows: $g(\theta) = f(\theta) + \frac{1}{2}\theta^T R \theta$ and perturbing f with Gaussian noise. Optimization problems of this form are very common in machine learning (where R corresponds to a regularizer) and control (where R corresponds to a control cost). The convexified optimization problem is:

$$\min_{\theta \in \mathcal{C}} f_\alpha(\theta) + \frac{1}{2}\theta^T R \theta \quad (2)$$

where $R \succeq 0$ and

$$f_\alpha(\theta) = \frac{1}{\alpha} \log \left(E_{\omega \sim \mathcal{N}(0, \Sigma)} [\exp(\alpha f(\theta + \omega))] \right).$$

This kind of objective is common in risk-averse optimization. To a first order Taylor expansion in α , the above objective is equal to $E[f(\theta + \omega)] + \alpha \text{Var}(f(\theta + \omega))$, indicating that increasing α will make the solution more robust to Gaussian perturbations. α is called the risk-factor and is a measure of the risk-aversion of the decision maker. Larger values of α will reject solutions that are not robust to Gaussian perturbations.

In order that the expectation exists, we require that f is bounded above:

$$\text{For all } \theta \in \mathbf{R}^k, f(\theta) \leq M < \infty. \quad (3)$$

We implicitly make this assumption throughout this paper in all the stated results. Note that this is not a very restrictive assumption, since, given any function g with a finite minimum, one can define a new objective $g' = \min(g, \bar{m})$, where \bar{m} is an upper bound on the minimum (say the value of the function at some point), without changing the minimum. Since the convex quadratic is non-negative, f is also bounded above by \bar{m} and hence $0 < \exp(\alpha f(\theta + \omega)) \leq \exp(\alpha \bar{m})$. This ensures that $f_\alpha(\theta)$ is finite. Some results will require differentiability, and we can preserve this by defining g' using a soft-min: For example, $g'(x) = \bar{m} \tanh\left(\frac{g(x)}{\bar{m}}\right)$.

Theorem 3.1 (Universal Convexification). *The optimization problem (2) is a convex optimization problem whenever \mathcal{C} is a convex set and $\alpha R \succeq \Sigma^{-1}$.*

Proof. Writing out the objective function in (2) and scaling by α , we get:

$$\log \left(E_{\omega \sim \mathcal{N}(0, \Sigma)} [\exp(\alpha f(\theta + \omega))] \right) + \frac{\alpha}{2} \theta^T R \theta \quad (4)$$

Writing out the expectation, we have:

$$\begin{aligned} E_{\omega \sim \mathcal{N}(0, \Sigma)} [\exp(\alpha f(\theta + \omega))] &= E_{y \sim \mathcal{N}(\theta, \Sigma)} [\exp(\alpha f(y))] \\ &\propto \int \exp\left(-\frac{1}{2}(y - \theta)^T \Sigma^{-1} (y - \theta)\right) \exp(\alpha f(y)) dy \end{aligned}$$

where we omitted the normalizing constant $\left(\sqrt{(2\pi)^n \det(\Sigma)}\right)^{-1}$ in the last step. Thus, exponentiating (4), we get (omitting the normalizing constant):

$$\int \exp\left(\frac{\alpha}{2} \theta^T R \theta - \frac{1}{2}(y - \theta)^T \Sigma^{-1} (y - \theta) + \alpha f(y)\right) dy$$

The term inside the exponent can be rewritten as

$$\frac{1}{2} \theta^T (\alpha R - \Sigma^{-1}) \theta + \theta^T \Sigma^{-1} y + \alpha f(y) - \frac{1}{2} y^T \Sigma^{-1} y$$

Since $\alpha R \succeq \Sigma^{-1}$, this is a convex quadratic function of θ for each y . Thus, the overall objective is the composition of a convex and increasing function $\log E[\exp(\cdot)]$ and a convex quadratic and is hence convex [Boyd and Vandenberghe, 2004]. Since \mathcal{C} is convex, the overall problem is a convex optimization problem. \square

3.1 INTERPRETATION

Theorem 3.1 is a surprising result, since the condition for convexity does not depend in any way on the

properties of f (except for the boundedness assumption (3)), but only on the relationship between the quadratic objective R , the risk factor α and the noise level Σ . In this section, we give some intuition behind the result and describe why it is plausible that this is true.

In general, arbitrary nonconvex optimization problems can be very challenging to solve. As a worst case example, consider a convex quadratic function $g(x) = x^2$ that is perturbed slightly: At some point where the function value is very large (say $x = 100$), we modify the function so that it suddenly drops to a large negative value (lower than the global minimum 0 of the convex quadratic). By doing this perturbation over a small finite interval, one can preserve differentiability while introducing a new global minimum far away from the original global minimum. In this way, one can create difficult optimization problems that cannot be solved using gradient descent methods, unless initialized very carefully.

The work we present here does not solve this problem: In fact, it will not find a global minimum of the form created above. The risk-aversion introduced destroys this minimum, since small perturbations cause the function to increase rapidly, ie, $g(\theta^* + \omega) \gg g(\theta^*)$, so that the objective (2) becomes large. Instead, it will find a “robust” minimum, in the sense that Gaussian perturbations around the minimum do not increase the value of the objective by much. This intuition is formalized by theorem 3.2, which bounds the suboptimality of the convexified solution relative to the optimal solution of the original problem in terms of the sensitivity of f to Gaussian perturbations around the optimum.

In figure 1, we illustrate the effect of the convexification for a 1-dimensional optimization problem. The blue curve represents the original function $g(\theta)$. It has 4-local minima in the interval $(-3, 3)$. Two of the shallow minima are eliminated by smoothing with Gaussian noise to get $\bar{g}(\theta)$. However, there is a deep but narrow local minimum that remains even after smoothing. Making the problem convex using risk-aversion and theorem 3.1 leads to the green curve that only preserves the robust minimum as the unique global optimum.

3.2 ANALYSIS OF SUB-OPTIMALITY

We have derived a convex surrogate for a very general class of optimization problems. However, it is possible that the solution to the convexified problem is drastically different from that of the original problem and the convex surrogate we propose is a poor approximation. Given the hardness of general non-convex opti-

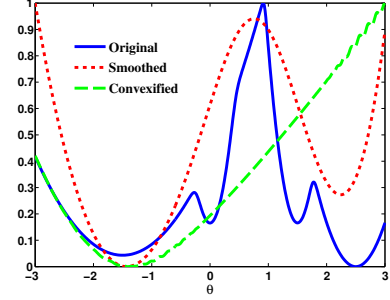


Figure 1: Illustration of Convexification for a 1-dimensional optimization problem

mization, we do not expect the two problems to have close solutions without additional assumptions. In this section, we analyze the gap between the original and convexified problem. In order to answer this, we first define the sensitivity function, which quantifies the gap between (1) and (2).

Definition 1 (Sensitivity Function). The sensitivity function of f at noise level Σ , risk α is defined as:

$$\begin{aligned} S_{\alpha, f}(\theta) &= \frac{1}{\alpha} \log \left(\mathbb{E} \left[\exp \left(\alpha \left(f(\theta + \omega) - \bar{f}(\theta) \right) \right) \right] \right) \\ &= \frac{1}{\alpha} \log \left(\mathbb{E} \left[\exp \left(\alpha \tilde{f}_{\omega}(\theta) \right) \right] \right). \end{aligned}$$

This is a measure of how sensitive f is to Gaussian perturbations at θ .

Theorem 3.2 (Suboptimality Analysis). Let $g(\theta) = f(\theta) + \frac{1}{2}\theta^T R \theta$ and define:

$$\theta_{\alpha}^* = \operatorname{argmin}_{\theta \in \mathcal{C}} f_{\alpha}(\theta) + \frac{1}{2}\theta^T R \theta, \quad \theta^* = \operatorname{argmin}_{\theta \in \mathcal{C}} \bar{g}(\theta).$$

Then,

$$\bar{g}(\theta_{\alpha}^*) - \bar{g}(\theta^*) \leq S_{\alpha, f}(\theta)$$

Proof. We have that $\forall \theta$, $\bar{g}(\theta) = \bar{f}(\theta) + \frac{1}{2}\theta^T R \theta + \frac{1}{2}\operatorname{tr}(\Sigma R)$. From the convexity of the function $y \rightarrow \exp(\alpha y)$ and Jensen’s inequality, we have

$$\begin{aligned} \bar{g}(\theta_{\alpha}^*) &= \bar{f}(\theta_{\alpha}^*) + \frac{1}{2}\theta_{\alpha}^{*T} R \theta_{\alpha}^* + \frac{1}{2}\operatorname{tr}(\Sigma R) \\ &\leq f_{\alpha}(\theta_{\alpha}^*) + \frac{1}{2}\theta_{\alpha}^{*T} R \theta_{\alpha}^* + \frac{1}{2}\operatorname{tr}(\Sigma R) \end{aligned}$$

Since $\theta_{\alpha}^* = \operatorname{argmin}_{\theta \in \mathcal{C}} f_{\alpha}(\theta) + \frac{1}{2}\theta^T R \theta$, for any $\theta \in \mathcal{C}$, we have

$$\begin{aligned} \bar{g}(\theta_{\alpha}^*) &\leq f_{\alpha}(\theta) + \frac{1}{2}\theta^T R \theta + \frac{1}{2}\operatorname{tr}(\Sigma R) \\ &= S_{\alpha, f}(\theta) + \bar{g}(\theta). \end{aligned}$$

The result follows by plugging in $\theta = \theta^*$ and subtracting $\bar{g}(\theta^*)$ from both sides. \square

Remark 1. Although we only prove suboptimality relative to the optimal solution of a smoothed version of g , we can extend the analysis to the optimal solution of g itself. Define $\theta^* = \operatorname{argmin}_{\theta \in \mathcal{C}} g(\theta)$. We can prove that

$$g(\theta_\alpha^*) - g(\theta^*) \leq (\bar{g}(\theta^*) - g(\theta^*)) + (g(\theta_\alpha^*) - \bar{g}(\theta_\alpha^*)) + S_{\alpha, f}(\theta)$$

Assuming that g changes slowly around θ^* and θ_α^* (indicative of the fact that θ^* is a “robust” minimum and θ_α^* is the minimum of a robustified problem), we can bound the first term. We leave a precise analysis for future work.

3.2.1 BOUNDING THE SENSITIVITY FUNCTION

The sensitivity function is exactly the moment generating function of the 0-mean random variable $f_\omega(\theta)$. Several techniques have been developed for bounding moment generation functions in the field of concentration inequalities [Boucheron et al., 2013]. Using these techniques, we can bound the moment generating function (i.e. the sensitivity function) under the assumption that f is Lipschitz-continuous. Before stating the lemma, we state a classical result:

Theorem 3.3 (Log-Sobolev Inequality, [Boucheron et al., 2013], theorem 5.4). *Let $\omega \sim \mathcal{N}(0, \sigma^2 I)$ and f be any continuously differentiable function of ω . Then,*

$$\mathbb{E} \left[f(\omega)^2 \log \left(\frac{f(\omega)^2}{\mathbb{E}[f(\omega)^2]} \right) \right] \leq 2\sigma^2 \mathbb{E} [\|\nabla f(\omega)\|^2].$$

Further, if f is Lipschitz with Lipschitz constant L , then

$$\mathbb{E} \left[\exp \left(\alpha \left(f(\omega) - \mathbb{E}[f(\omega)] \right) \right) \right] \leq \exp(\alpha^2 L^2 \sigma^2 / 2).$$

Finally, we can also prove that if f is differentiable and $\mathbb{E} [\exp \lambda \|\nabla f(\omega)\|^2] < \infty$ for all $\lambda < \lambda_0$, then, given η such that $\lambda \sigma^2 < \eta \lambda_0$ and $\lambda \eta < 2$, we have

$$\log \left(\mathbb{E} \left[\exp \left(\lambda \left(f(\omega) - \mathbb{E}[f(\omega)] \right) \right) \right] \right) \leq \frac{\lambda \eta}{2 - \lambda \eta} \log \left(\mathbb{E} \left[\exp \left(\frac{\lambda \sigma^2}{\eta} \|\nabla f(\omega)\|^2 \right) \right] \right)$$

Theorem 3.4 (Lipschitz Continuous Functions). *Assume the same setup as theorem 3.2. Suppose further that f is Lipschitz continuous with Lipschitz constant L , that is,*

$$\forall \theta, \theta', |f(\theta) - f(\theta')| \leq L \|\theta - \theta'\|.$$

Then, $S_{\alpha, f}(\theta) \leq \frac{1}{2} \alpha L^2 \lambda_{\max}(\Sigma)$. Further,

$$\bar{g}(\theta) - \min_{\theta \in \mathcal{C}} \bar{g}(\theta) \leq \frac{\alpha L^2 \lambda_{\max}(\Sigma)}{2}.$$

Proof. We can write

$$\tilde{f}(\omega') = f(\theta + \Sigma^{1/2} \omega') - \bar{f}(\theta)$$

where $\omega' \sim \mathcal{N}(0, I_k)$.

$$\begin{aligned} |\tilde{f}(\omega'_1) - \tilde{f}(\omega'_2)| &= |f(\theta + \Sigma^{1/2} \omega'_1) - f(\theta + \Sigma^{1/2} \omega'_2)| \\ &\leq L \|\Sigma^{1/2}(\omega'_1 - \omega'_2)\| \leq L \lambda_{\max}(\Sigma^{1/2}) \|\omega'_1 - \omega'_2\| \end{aligned}$$

Since $\lambda_{\max}(\Sigma^{1/2}) = \sqrt{\lambda_{\max}(\Sigma)}$, this shows that \tilde{f} is Lipschitz with Lipschitz constant $\sqrt{\lambda_{\max}(\Sigma)}L$. The result now follows from theorem 3.3. From theorem 3.2, this implies that

$$\bar{g}(\theta) - \min_{\theta \in \mathcal{C}} \bar{g}(\theta) \leq \frac{\alpha L^2 \lambda_{\max}(\Sigma)}{2}.$$

□

4 ALGORITHMS AND CONVERGENCE GUARANTEES

In general, the expectations involved in (2) cannot be computed analytically. Thus, we need to resort to sampling based approaches in order to solve these problems. This has been studied extensively in recent years in the context of machine learning, where stochastic gradient methods and variants have been shown to be efficient, particularly in the context training machine learning algorithms with huge amounts of data. We now describe stochastic gradient methods for solving (2) and adapt the convergence guarantees of stochastic gradient methods to our setting.

4.1 Stochastic Gradient Methods with Convergence Guarantees

In this section, we will derive gradients of the convex objective function (2). We will assume that the function f is differentiable at all $\theta \in \mathbf{R}^k$. In order to get unbiased gradient estimates, we exponentiate the objective (2) to get:

$$\mathbf{G}(\theta) = \mathbb{E}_{\omega \sim \mathcal{N}(0, \Sigma)} \left[\exp \left(\alpha f(\theta + \omega) + \frac{\alpha}{2} \theta^T R \theta \right) \right].$$

Since $f(\theta)$ is differentiable for all θ , so is $\exp(\alpha f(\theta + \omega) + \frac{1}{2} \theta^T R \theta)$. Further, suppose that

$$\mathbb{E} \left[\exp(2\alpha f(\theta + \omega)) \|\nabla f(\theta + \omega) + R\theta\|^2 \right]$$

exists and is finite for each θ . Then, if we differentiate $\mathbf{G}(\theta)$ with respect to θ , we can interchange the expectation and differentiation to get

$$\mathbb{E} \left[\exp \left(\alpha f(\theta + \omega) + \frac{\alpha}{2} \theta^T R \theta \right) (\alpha \nabla f(\theta + \omega) + \alpha R \theta) \right]$$

Thus, we can sample $\omega \sim \mathcal{N}(0, \Sigma)$ and get an unbiased estimate of the gradient

$$\alpha \exp \left(\alpha f(\theta + \omega) + \frac{\alpha}{2} \theta^T R \theta \right) (\nabla f(\theta + \omega) + R\theta) \quad (5)$$

which we denote by $\hat{\nabla} \mathbf{G}(\theta, \omega)$. As in standard stochastic gradient methods, one saves on the complexity of a single iteration by using a single (or a small number of) samples to get a gradient estimate while still converging to the global optimum with high probability and in expectation, because over multiple iterations one moves along the negative gradient “on average”.

Algorithm 1 Stochastic Gradient Method for (2)

$\theta \leftarrow 0$
for $i = 1, \dots, T$ **do**
 $\omega \sim \mathcal{N}(0, \Sigma), \theta \leftarrow \text{Proj}_{\mathcal{C}} \left(\theta - \eta_i \hat{\nabla} \mathbf{G}(\theta, \omega) \right)$
end for

From standard convergence theory for stochastic gradient methods [Bubeck, 2013], we have:

Corollary 1. *Suppose that $\mathbb{E} \left[\left\| \hat{\nabla} \mathbf{G}(\theta, \omega) \right\|^2 \right] \leq \zeta^2$, \mathcal{C} is contained in a ball of radius $R(\mathcal{C})$ and $\alpha R \succeq \Sigma^{-1}$. Run algorithm 1 for T iterations with $\eta_i = \frac{R(\mathcal{C})}{\zeta} \sqrt{\frac{1}{2i}}$ and define $\hat{\theta} = \frac{1}{T} \sum_{i=1}^T \theta_i$, $\mathbf{G}^* = \arg \min_{\theta \in \mathcal{C}} \mathbf{G}(\theta)$. Then, we have*

$$\mathbb{E} \left[\mathbf{G}(\hat{\theta}) \right] - \mathbf{G}^* \leq R(\mathcal{C}) \zeta \sqrt{\frac{1}{2T}}$$

In the following theorem, we prove a convergence rate for algorithm 1.

Theorem 4.1. *Suppose that $R = \kappa I, \Sigma = \sigma^2 I, \alpha \kappa \succeq \frac{1}{\sigma^2}$, \mathcal{C} is contained in a sphere of radius $R(\mathcal{C})$ and that for all $\theta \in \mathcal{C}$ $\bar{g}(\theta) \leq \bar{m}$. Also, suppose that for some $\beta < (\alpha)^{-1}$:*

$$\frac{1}{2\alpha} \log \left(\mathbb{E}_{\omega \sim \mathcal{N}(0, \Sigma)} \left[\exp \left(2 \frac{\alpha \sigma^2}{\beta} \left\| \nabla f(\theta + \omega) \right\|^2 \right) \right] \right) \leq \gamma^2$$

Define $\delta = \sqrt{\frac{\beta \gamma^2}{\sigma^2(1-\alpha\beta)}} + \kappa R(\mathcal{C})$. Then, we can choose $\zeta \leq \alpha^2 \delta^2 \exp \left(2\alpha(\bar{m} + \gamma^2) + \frac{\alpha\beta}{1-\alpha\beta} - \sigma^2 \kappa \right)$.

Remark 2. The convergence guarantees are in terms of the exponentiated objective $\mathbf{G}(\theta)$. We can convert these into bounds on $\log(\mathbf{G}(\theta))$ as follows:

$$\begin{aligned} \mathbb{E} \left[\log(\mathbf{G}(\hat{\theta})) \right] &\leq \log \left(\mathbb{E} \left[\mathbf{G}(\hat{\theta}) \right] \right) \leq \\ &\log \left(\mathbf{G}^* + \frac{\zeta R(\mathcal{C})}{\sqrt{2T}} \right) \end{aligned}$$

where the first inequality follows from concavity of the log function. Subtracting $\log(\mathbf{G}^*)$, we get

$$\mathbb{E} \left[\log(\mathbf{G}(\hat{\theta})) \right] - \log(\mathbf{G}^*) \leq \log \left(1 + \frac{\zeta R(\mathcal{C})}{\sqrt{2T} \mathbf{G}^*} \right).$$

5 CONTROL PROBLEMS

In this section, we extend the above approach to the control of discrete-time dynamical systems. Stochastic optimal control of nonlinear systems in general is a hard problem and the only known general approach is based on dynamic programming, which scales exponentially with the dimension of the state space. Algorithms that approximate the solution of the dynamic program directly (approximate dynamic programming) have been successful in various domains, but scaling these approaches to high dimensional continuous state control problems has been challenging. In this section, we pursue the alternate approach of policy search or policy gradient methods [Baxter and Bartlett, 2001]. These algorithms have the advantage that they are directly optimizing the performance of a control policy as opposed to a surrogate measure like the error in the solution to the Bellman equation. They have been used successfully for various applications and are closely related to path integral control [Kappen, 2005, Theodorou et al., 2010b]. However, in all of these approaches, there were no guarantees made regarding the optimality of the policy that the algorithm converges to (even in the limit of infinite sampling) or the rate of convergence.

In this work, we develop the *first* policy gradient algorithms that achieve the *globally* optimal solutions to a class of *risk-averse* policy optimization problems.

5.1 Problem Setup

We deal with arbitrary discrete-time dynamical systems of the form

$$\begin{aligned} \epsilon &= (\epsilon_1 \quad \dots \quad \epsilon_{N-1})^T \sim \mathbb{P}_\epsilon \\ x_1 &= 0, \quad x_{t+1} = \mathcal{F}(x_t, y_t, \epsilon_t, t) \quad t = 1, \dots, N-1 \end{aligned} \quad (6)$$

$$y_t = u_t + \omega_t, \quad \omega_t \sim \mathcal{N}(0, \Sigma_t) \quad t = 1, \dots, N-1 \quad (7)$$

where $x_t \in \mathbf{R}^{n_s}$ denotes the state, $y_t \in \mathbf{R}^{n_u}$ the effective control input, $\epsilon_t \in \mathbf{R}^p$ external disturbances, $u_t \in \mathbf{R}^{n_u}$ the actual control input, $\omega_t \in \mathbf{R}^{n_u}$ the control noise, $\mathcal{F} : \mathbf{R}^{n_s} \times \mathbf{R}^{n_u} \times \mathbf{R}^p \times \{1, \dots, N-1\} \mapsto \mathbf{R}^{n_s}$ the discrete-time dynamics. In this section, we will use boldface to denote quantities stacked over time (like ϵ). Equation (6) can model any noisy discrete-time dynamical system, since \mathcal{F} can be any function of the current state, control input and external disturbance

(noise). However, we require that all the control dimensions are affected by Gaussian noise as in (7). This can be thought of either as real actuator noise or artificial exploration noise. The choice of zero initial state $x_1 = 0$ is arbitrary - our results even extend to an arbitrary distribution over the initial state.

We will work with costs that are a combination of arbitrary state costs and quadratic control costs:

$$\sum_{t=1}^N \ell_t(x_t) + \sum_{t=0}^{N-1} \frac{u_t^T R_t u_t}{2} \quad (8)$$

where $\ell_t(x_t)$ is the stage-wise state cost at time t . ℓ_t can be any bounded function of the state-vector x_t . Further, we will assume that the control-noise is non-degenerate, that is Σ_t is full rank for all $0 \leq t \leq N-1$. We denote $S_t = \Sigma_t^{-1}$. We seek to design feedback policies

$$u_t = K_t \phi(x_t, t), \phi : \mathbf{R}^{n_s} \times \{1, 2, \dots, N-1\} \mapsto \mathbf{R}^r$$

$$K_t \in \mathbf{R}^{n_u \times r} \quad (9)$$

to minimize the accumulated cost (8). We will assume that the features ϕ are fixed and we seek to optimize the policy parameters $\mathbf{K} = \{K_t : t = 1, 2, \dots, N-1\}$. The stochastic optimal control problem we consider is defined as follows:

$$\begin{aligned} & \underset{\mathbf{K}}{\text{Minimize}} \quad \mathbb{E}_{\epsilon, \omega_t} [\exp(\alpha \mathcal{L}(\mathbf{K}))] \\ & \text{Subject to } x_1 = 0, x_{t+1} = \mathcal{F}(x_t, y_t, \epsilon_t, t) \\ & \quad y_t = u_t + \omega_t, u_t = K_t \phi(x_t, t) \\ & \quad \epsilon \sim \mathbb{P}_\epsilon, \omega_t \sim \mathcal{N}(0, \Sigma_t) \\ & \quad \mathcal{L}(\mathbf{K}) = \sum_{t=0}^N \ell_t(x_t) + \sum_{t=0}^{N-1} \frac{u_t^T R_t u_t}{2} \end{aligned} \quad (10)$$

This is exactly the same as the formulation in Risk Sensitive Markov Decision Processes [Marcus et al., 1997], the only change being that we have explicitly separated the noise appearing in the controls from the noise in the dynamical system overall. In this formulation, the objective depends not only on the average behavior of the control policy but also on variance and higher moments (the tails of the distribution of costs). This has been studied for linear systems under the name of LEQG control [Speyer et al., 1974]. α is called the risk factor: Large positive values of α result in strongly risk-averse policies while large negative values result in risk-seeking policies. In our formulation, we will need a certain minimum degree of risk-aversion for the resulting policy optimization problem to be convex.

5.2 Convex Controller Synthesis

Theorem 5.1. *If $\alpha R_t \succeq (\Sigma_t)^{-1} = S_t$ for $t = 1, \dots, N-1$, then the optimization problem (10) is*

convex.

Proof. We first show that for a fixed ϵ , the quantity $\mathbb{E}_{\omega_t \sim \mathcal{N}(0, \Sigma_t)} [\exp(\alpha \mathcal{L}(\mathbf{K}))]$ is a convex function of \mathbf{K} . Then, by the linearity of expectation, so is the original objective. We can write down the above expectation (omitting the normalizing constant of the Gaussian) as:

$$\int \exp \left(- \sum_{t=1}^{N-1} \frac{1}{2} \|y_t - K_t \phi(x_t, t)\|_{S_t}^2 + \alpha \mathcal{L}(\mathbf{K}) \right) dy$$

If we fix \mathbf{y}, ϵ , using (6), we can construct x_t for every $t = 1, \dots, N$. Thus, \mathbf{x} is a deterministic function of \mathbf{y}, ϵ and does not depend on \mathbf{K} . The term inside the exponential can be written as

$$\begin{aligned} & - \frac{1}{2} \left(\sum_{t=1}^{N-1} \|y_t\|_{S_t}^2 \right) + \alpha \left(\sum_{t=1}^N \ell_t(x_t) \right) \\ & + \sum_{t=1}^{N-1} \frac{1}{2} \text{tr} \left((K_t^T (\alpha R_t - S_t) K_t) \phi(x_t, t) \phi(x_t, t)^T \right) \\ & - \sum_{t=1}^{N-1} y_t^T S_t \phi(x_t, t) K_t \end{aligned}$$

The terms on the first line don't depend on \mathbf{K} . The function $(K_t^T (\alpha R_t - S_t) K_t)$ is convex in \mathbf{K} with respect to the semidefinite cone [Boyd and Vandenberghe, 2004] when $\alpha R_t - S_t \succeq 0$ and $\phi(x_t, t) \phi(x_t, t)^T$ is a positive semidefinite matrix. Hence the term on the second line is convex in \mathbf{K} . The term on the third line is linear in \mathbf{K} and hence convex. Since \exp is a convex and increasing function, the composed function (which is the integrand) is convex as well in \mathbf{K} . Thus, the integral is convex in \mathbf{K} . \square

We can add arbitrary convex constraints and penalties on \mathbf{K} without affecting convexity.

Corollary 2. *The problem*

$$\begin{aligned} & \min_{\mathbf{K}} \quad \mathbb{E}_{\epsilon \sim \mathbb{P}_\epsilon, \omega_t \sim \mathcal{N}(0, \Sigma_t)} [\exp(\alpha \mathcal{L}(\mathbf{K}))] \\ & \text{Subject to} \quad (6), (7), \mathbf{K} \in \mathcal{C} \end{aligned} \quad (11)$$

is a convex optimization problem for any arbitrary convex set $\mathcal{C} \subset \mathbf{R}^{n_u \times r \times (N-1)}$ if $\alpha R_t \succeq S_t \quad \forall t$.

6 NUMERICAL RESULTS

In this section, we present preliminary numerical results illustrating applications of the framework to various problems with comparisons to a simple baseline approach. These are not meant to be thorough numerical comparisons but simple illustrations of the power and applications of our framework.

6.1 BINARY CLASSIFICATION

We look at a problem of binary classification. Let y denote the actual label and \hat{y} denote the predicted label. We use the loss function

$$\ell(y, \hat{y}) = \begin{cases} -\infty & \text{if } y\hat{y} > 0 \\ 0 & \text{otherwise} \end{cases}.$$

This is a non-convex loss function (the logarithm of the standard 0-1 loss). We convexify this in the prediction \hat{y} using our approach:

$$\frac{1}{\alpha} \log \left(\mathbb{E} [\exp(\alpha \ell(y, \hat{y}))] \right) + \frac{(\hat{y})^2}{2\sigma^2}.$$

Plugging in $\hat{y} = \theta^T x$ where x is the feature vector, we get

$$\frac{1}{\alpha} \log \left(\mathbb{E}_{\omega \sim \mathcal{N}(0, \sigma^2)} [\exp(\alpha \ell(y, \theta^T x + \omega))] \right) + \frac{(\theta^T x)^2}{2\alpha\sigma^2}.$$

Plugging in the expression for ℓ gives

$$\frac{1}{\alpha} \log \left(\frac{1}{2} \operatorname{erfc} \left(\frac{y\theta^T x}{\sqrt{2}\sigma} \right) \right) + \frac{(\theta^T x)^2}{2\alpha\sigma^2}.$$

where erfc is the Gaussian error function. Given a dataset $\{(x_i, y_i)\}$, we can form the empirical risk-minimization problem with this convexified objective:

$$\sum_{i=1}^M \frac{1}{\alpha} \log \left(\frac{1}{2} \operatorname{erfc} \left(\frac{y_i \theta^T x_i}{\sqrt{2}\sigma} \right) \right) + \frac{(\theta^T x_i)^2}{2\alpha\sigma^2}$$

We can drop the α since it only scales the objective (this is a consequence of the fact that $\exp(\ell)$ is 0-1 valued and does not change on raising it to a positive power). Thus, we finally end up with

$$\frac{1}{M} \sum_{i=1}^M \left(\log \left(\frac{1}{2} \operatorname{erfc} \left(\frac{y_i \theta^T x_i}{\sqrt{2}\sigma} \right) \right) + \frac{(\theta^T x_i)^2}{2\sigma^2} \right).$$

The first term is a data-fit term (a smoothed version of the 0-1 loss) and the second term is a regularizer, although we penalize the prediction $\theta^T x$ rather than θ itself. If x are normalized and span all directions, by summing over the entire dataset we get something close to the standard Tikhonov regularization.

We compare the performance of our convexification-based approach with a standard implementation of a Support Vector Machine (SVM) [Chang and Lin, 2011]. We use the breast cancer dataset from [Bache and Lichman, 2013]. We compare the two algorithms on various train-test splits of the dataset (without using cross-validation or parameter tuning). For each split, we create a noisy version of the dataset by

adding Gaussian noise to the labels and truncating to $+1/-1$: $\hat{y}^i = \operatorname{sign}(y^i + \omega)$, $\omega \sim \mathcal{N}(0, \sigma^2)$. The accuracy of the learned classifiers on withheld test-data, averaged over 50 random train-test splits with label corruption as described above, are plotted as function of the noise level σ in figure 1. This is not a completely fair comparison since our approach explicitly optimizes for the worst case under Gaussian perturbations to the prediction (which can also be seen as a Gaussian perturbation to the label). However, as mentioned earlier, the purpose of these numerical experiments is to illustrate the applicability of our convexification approach to various problems so we do not do a careful comparison to robust variants of SVMs, which would be better suited to the setting described here.

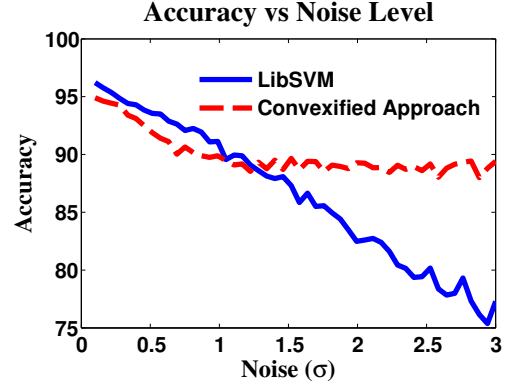


Figure 2: Binary Classification

6.2 CLASSIFICATION WITH NEURAL NETWORKS

We present an algorithm that does neural network training using the results of section 5. Each layer of the neural network is a time-step in a dynamical system, and the neural network weights correspond to the time-varying policy parameters. Let h denote a component-wise nonlinearity applied to its vector-input (a transfer function). The deterministic dynamics is

$$x_{t+1} = h(K_t x_t), x_0 = x$$

where x_t is the vector of activations at the t -th layer, K_t is the weight matrix and x is the input to the neural network. The output is x_N , where N is the number of layers in the network. The cost function is simply the loss function between the output of the neural network x_N and a desired output y : $\ell(y, x_N)$. To put this into our framework, we add noise to the input of the transfer function at each layer:

$$x_{t+1} = h(K_t x_t + \omega_t), x_1 = x, \omega_t \sim \mathcal{N}(0, \Sigma_t)$$

Additionally, we define the objective to be

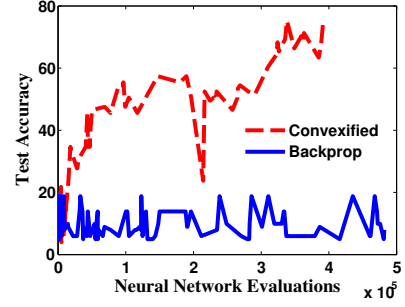
$$\mathbb{E} \left[\exp \left(\alpha \ell(y, x_N) + \sum_{t=1}^{N-1} \frac{\|K_t x_t\|_{\Sigma_{t-1}}^2}{2} \right) \right]$$

where the expectation is with respect to the Gaussian noise added at each layer in the network. Note that the above objective is a function of \mathbf{K}, x, y . The quadratic penalty on $K_t x_t$ can again be thought of as a particular type of regularization which encourages learning networks with small internal activations. We add this objective over the entire dataset $\{x^i, y^i\}$ to get our overall training objective.

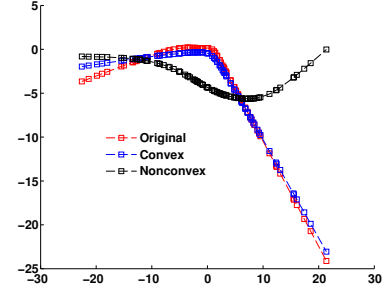
We evaluate this approach on a small randomly selected subset of the MNIST dataset [LeCun et al., 1998]. We use the version available at <http://nicolas.le-roux.name/> along with the MATLAB code provided for training neural networks. We use a 2-layer neural network with 20 units in the hidden layer and tanh-transfer functions in both layers. We use a randomly chosen collection of 900 data points for training and another 100 data points for validation. We compare training using our approach with simple backprop based training. Both of the approaches use a stochastic gradient - in our approach the stochasticity is both in selection of the data point i and the realization of the Gaussian noise ω while in standard backprop the stochasticity is only in the selection of i . We plot learning curves (in terms of generalization or test error) for both approaches, as a function of the number of neural network evaluations (forward+back prop) performed by the algorithm in figure 3a. The nonconvex approach based on standard backprop-gradient descent gets stuck in a local minimum and does not improve test accuracy much. On the other hand, the convexified approach is able to learn a classifier that generalizes better. We also compared backprop with training a neural network on a 1-dimensional regression problem where the red curve represents the original function with data-points indicated by squares, the blue curve the reconstruction learned by our convexified training approach and the black curve the reconstruction obtained by using backprop (figure 3b). Again, backprop gets stuck in a bad local minimum while our approach is able to find a fairly accurate reconstruction.

7 CONCLUSION AND FUTURE WORK

We have developed a general framework for convexifying a broad class of optimization problems, analysis that relates the solution of the convexified problem to the original one and given algorithms with convergence rate guarantees to solve the convexified problems. Ex-



(a) Classification



(b) Regression

Figure 3: Training Neural Networks

tending the framework to dynamical systems, we derive the first approach to policy optimization with optimality and convergence rate guarantees. We validated our approach numerically on problems of binary classification and training neural networks. In future work, we will refine the suboptimality analysis for our convexification approach. Algorithmically, stochastic gradient methods could be slow if the variance in the gradient estimates is high, which is the case when using the exponentiated objective (as in section 4). We will study the applicability of recent work on using better sampling algorithms with stochastic gradient [Atchade et al., 2014] to our convexified problems.

APPENDIX

Corollary 3. [Boucheron et al., 2013], corollary 4.15 Let P, Q be arbitrary distributions on some space Ω and $f : \Omega \rightarrow \mathbf{R}$ is such that $\mathbb{E}_{\omega \sim P} [\exp(f(\omega))] < \infty$. Then, the Kullback-Leibler divergence satisfies:

$$\text{KL}(Q \parallel P) = \sup_f \left[\mathbb{E}_{\omega \sim Q} [f(\omega)] - \log \left(\mathbb{E}_{\omega \sim P} [\exp(f(\omega))] \right) \right]$$

7.1 PROOF OF THEOREM 4.1

Throughout this section, Let $\tilde{\alpha} = 2\alpha$ and P denote the Gaussian density $\mathcal{N}(0, \Sigma)$. Define a new distribution Q with density $Q(\omega) \propto P(\omega) \exp(\tilde{\alpha} \tilde{f}_\omega(\theta))$. We denote $\hat{\nabla} \mathbf{G}(\theta, \omega)$ by $\hat{\nabla} \mathbf{G}$, $\nabla f(\theta + \omega)$ by $\hat{\nabla} f$ and

$\nabla f(\theta + \omega) + \kappa\theta$ by $\tilde{\nabla}f$ for brevity. Expectations are always with respect to $\omega \sim P$, unless denoted otherwise.

Proof. $(\alpha)^{-2} \mathbb{E} \left[\left\| \hat{\nabla} \mathbf{G} \right\|^2 \right]$ evaluates to

$$\begin{aligned} & \exp(\alpha\kappa\theta^T\theta) \mathbb{E} \left[\exp(\tilde{\alpha}f(\omega + \theta)) \left\| \tilde{\nabla}f \right\|^2 \right] = \\ & \exp\left(\tilde{\alpha} \left(\frac{\kappa}{2}\theta^T\theta + \bar{f}(\theta) \right)\right) \mathbb{E} \left[\exp\left(\tilde{\alpha}\tilde{f}_\omega(\theta)\right) \left\| \tilde{\nabla}f \right\|^2 \right] = \\ & \exp\left(\tilde{\alpha} \left(\bar{g}(\theta) - \frac{1}{2}\sigma^2\kappa \right)\right) \mathbb{E} \left[\exp\left(\tilde{\alpha}\tilde{f}_\omega(\theta)\right) \left\| \tilde{\nabla}f \right\|^2 \right] \end{aligned} \quad (12)$$

By the theorem hypotheses, the term outside the expectation is bounded above by $\alpha^2 \exp(2\alpha\bar{m} - \sigma^2\kappa)$. We are left with

$$\mathbb{E} \left[\exp\left(\tilde{\alpha}\tilde{f}_\omega(\theta)\right) \left\| \tilde{\nabla}f \right\|^2 \right].$$

Dividing this by $\mathbb{E} \left[\exp\left(\tilde{\alpha}\tilde{f}_\omega(\theta)\right) \right]$, we get

$$\frac{\mathbb{E} \left[\exp\left(\tilde{\alpha}\tilde{f}_\omega(\theta)\right) \left\| \tilde{\nabla}f \right\|^2 \right]}{\mathbb{E} \left[\exp\left(\tilde{\alpha}\tilde{f}_\omega(\theta)\right) \right]} = \mathbb{E}_{\omega \sim Q} \left[\left\| \tilde{\nabla}f \right\|^2 \right]. \quad (13)$$

Expanding $\left\| \tilde{\nabla}f \right\|^2$, we get

$$\begin{aligned} & \left\| \hat{\nabla}f \right\|^2 + \kappa^2 \|\theta\|^2 + 2\hat{\nabla}f^T\theta \\ & \leq \left\| \hat{\nabla}f \right\|^2 + \kappa^2 \mathbf{R}(\mathcal{C})^2 + 2\kappa \left\| \hat{\nabla}f \right\| \mathbf{R}(\mathcal{C}) \end{aligned}$$

Finally from lemma 1, we have

$$\mathbb{E}_{\omega \sim Q} \left[\left\| \hat{\nabla}f \right\|^2 \right] \leq \frac{\beta\gamma^2}{\sigma^2(1-\alpha\beta)}$$

and by concavity of the square-root function,

$$\mathbb{E}_{\omega \sim Q} \left[\left\| \hat{\nabla}f \right\| \right] \leq \sqrt{\frac{\beta\gamma^2}{\sigma^2(1-\alpha\beta)}}.$$

Plugging this bounds into the square expansion and letting $\delta = \sqrt{\frac{\beta\gamma^2}{\sigma^2(1-\alpha\beta)}} + \kappa\mathbf{R}(\mathcal{C})$, we get

$$\mathbb{E}_{\omega \sim Q} \left[\left\| \nabla f(\theta + \omega) + \kappa\theta \right\|^2 \right] \leq \delta^2 \quad (14)$$

From (12),(13) and (14), $\mathbb{E} \left[\left\| \hat{\nabla} \mathbf{G} \right\|^2 \right]$ is smaller than

$$\alpha^2 \exp(2\alpha\bar{m} - \sigma^2\kappa) \delta^2 \mathbb{E} \left[\exp\left(\tilde{\alpha}\tilde{f}_\omega(\theta)\right) \right]$$

Finally, by the last part of theorem 3.3,

$$\mathbb{E} \left[\exp\left(\tilde{\alpha}\tilde{f}_\omega(\theta)\right) \right] \leq \exp\left(\frac{\alpha\beta}{1-\alpha\beta} + 2\alpha\gamma^2\right).$$

Combining the two above results gives the theorem. \square

lemma 1. *Under the assumptions of theorem 4.1,*

$$\mathbb{E}_{\omega \sim Q} \left[\sigma^2 \left\| \nabla f(\theta + \omega) \right\|^2 \right] \leq \frac{\beta\gamma^2}{1-\alpha\beta} \quad (15)$$

Proof. The KL-divergence between Q and P is given by

$$\int \frac{P(\omega) \exp\left(\tilde{\alpha}\tilde{f}_\omega(\theta)\right)}{\mathbb{E} \left[\exp\left(\tilde{\alpha}\tilde{f}_\omega(\theta)\right) \right]} \log \left(\frac{\exp\left(\tilde{\alpha}\tilde{f}_\omega(\theta)\right)}{\mathbb{E} \left[\exp\left(\tilde{\alpha}\tilde{f}_\omega(\theta)\right) \right]} \right) d\omega.$$

By theorem 3.3 applied to $\exp\left(\frac{\tilde{\alpha}}{2}f(\theta + \omega)\right)$, the above quantity is bounded above by $\frac{1}{2}(\tilde{\alpha}\sigma)^2 \mathbb{E}_{\omega \sim Q} \left[\left\| \hat{\nabla}f \right\|^2 \right]$. Then, by corollary 3,

$$\begin{aligned} \text{KL}(Q \parallel P) & \geq \mathbb{E}_{\omega \sim Q} \left[\frac{\tilde{\alpha}\sigma^2}{\beta} \left\| \hat{\nabla}f \right\|^2 \right] \\ & \quad - \log \left(\mathbb{E} \left[\exp\left(\frac{\tilde{\alpha}\sigma^2}{\beta} \left\| \hat{\nabla}f \right\|^2\right) \right] \right). \end{aligned}$$

Denote the second term in the RHS by Γ . Plugging in the upper bound on $\text{KL}(Q \parallel P)$, we get

$$\Gamma \geq \sigma^2 \left(\frac{\tilde{\alpha}}{\beta} - \frac{\tilde{\alpha}^2}{2} \right) \mathbb{E}_{\omega \sim Q} \left[\left\| \nabla f(\theta + \omega) \right\|^2 \right].$$

Since the LHS is upper bounded by $\tilde{\alpha}\gamma^2$ (hypothesis of theorem) which gives us the bound

$$\mathbb{E}_{\omega \sim Q} \left[\sigma^2 \left\| \nabla f(\theta + \omega) \right\|^2 \right] \leq \frac{2\beta\gamma^2}{2 - \tilde{\alpha}\beta} = \frac{\beta\gamma^2}{1 - \alpha\beta}.$$

\square

Acknowledgements

This work was funded by the US National Science Foundation. We thank the anonymous UAI reviewers for valuable suggestions that greatly enhanced the readability of this paper. Extended versions of this paper will be posted at <http://arxiv.org/abs/1406.0554>

References

Y. F. Atchade, G. Fort, and E. Moulines. On stochastic proximal gradient algorithms. *ArXiv e-prints*, February 2014.

- K. Bache and M. Lichman. UCI machine learning repository, 2013. URL <http://archive.ics.uci.edu/ml>.
- J. Baxter and P. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350, 2001.
- Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration inequalities: A nonasymptotic theory of independence*. Oxford University Press, 2013.
- S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, Cambridge, UK, 2004.
- B. Van Den Broek, W. Wiegerinck, and B. Kappen. Risk sensitive path integral control. In *Uncertainty in AI, 2010. Proceedings of the 2010*, 2010.
- Sebastian Bubeck. The complexities of optimization, December 2013. URL <https://blogs.princeton.edu/imabandit/2013/04/25/orf523-noisy-oracles/>.
- Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- K. Dvijotham and E. Todorov. A unifying framework for linearly-solvable control. *Uncertainty in Artificial Intelligence*, 2011.
- W. Fleming and S. Mitter. Optimal control and nonlinear filtering for nondegenerate diffusion processes. *Stochastics*, 8:226–261, 1982.
- H.J. Kappen. Linear theory for control of nonlinear stochastic systems. *Physical Review Letters*, 95(20):200201, 2005.
- Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- S.I. Marcus, E. Fernández-Gaucherand, D. Hernández-Hernandez, S. Coraluppi, and P. Fard. Risk sensitive Markov decision processes. *Systems and Control in the Twenty-First Century*, 29, 1997.
- Hossein Mobahi. *Optimization by Gaussian Smoothing with Application to Geometric Alignment*. PhD thesis, University of Illinois at Urbana Champaign, Dec 2012.
- A. Rangarajan. Generalised graduated non-convexity algorithm for maximum a posteriori image estimation. In *Proc. ICPR*, pages 127–133, 1990.
- J. Speyer, John Deyst, and D. Jacobson. Optimization of stochastic linear systems with additive measurement and process noise using exponential performance criteria. *Automatic Control, IEEE Transactions on*, 19(4):358–366, 1974.
- E. Theodorou, J. Buchli, and S. Schaal. Reinforcement learning of motor skills in high dimensions: A path integral approach. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2397–2403. IEEE, 2010a.
- Evangelos Theodorou, Jonas Buchli, and Stefan Schaal. A generalized path integral control approach to reinforcement learning. *The Journal of Machine Learning Research*, 9999:3137–3181, 2010b.
- E. Todorov. Efficient computation of optimal actions. *Proceedings of the National Academy of Sciences*, 106(28):11478, 2009.
- M. Toussaint. Robot trajectory optimization using approximate inference. *International Conference on Machine Learning*, 26:1049–1056, 2009.

Structured Proportional Jump Processes

Tal El-Hay **Omer Weissbrod** **Elad Eban*** **Maurizio Zazzi** **Francesca Incardona**
Machine Learning for Healthcare Department of Medical Biotechnologies InformaPRO S.r.l.
and Life Sciences University of Siena, Italy EuResist Network GEIE, Italy
IBM Research – Haifa

Abstract

Learning the association between observed variables and future trajectories of continuous-time stochastic processes is a fundamental task in dynamic modeling. Often the dynamics are non-homogeneous and involve a large number of interacting components. We introduce a conditional probabilistic model that captures such dynamics, while maintaining scalability and providing an explicit way to express the interrelation between the system components. The principal idea is a factorization of the model into two distinct elements: one depends only on time and the other depends on the system configuration. We developed a learning procedure, given either full or point observations, and tested it on simulated data. We applied the proposed modeling scheme to study large cohorts of diabetes and HIV patients, and demonstrate that the factorization helps shed light on the dynamics of these diseases.

1 INTRODUCTION

Studies of dynamic systems often attempt to investigate the dependency of these dynamics on a set of static explanatory variables. In many cases, the studied process is composed of interrelated components that evolve continuously in time; hence, inter-component interactions are of interest as well. Examples appear in diverse fields, ranging from medicine to computational biology and economics.

Inferring such conditional dynamics of a real life system involves several challenges. We illustrate these challenges by our motivating example of studying disease progression in patients infected with *Human Immunodeficiency Virus*

* Currently at: The Selim and Rachel Benin School of Computer Science and Engineering. The Hebrew University of Jerusalem.

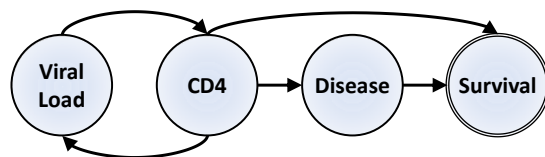


Figure 1: A graphical representation of a SCUP model for HIV. Directed arcs indicate directions of influence.

(HIV). The two common measures of the severity of HIV infection, viral load and the immune system CD4 protein count, are interrelated. A higher viral load weakens the immune system, while a weakened immune system potentially affects viral dynamics. A weakened immune system also increases the risk of death, either directly or by increasing the risk of contracting other diseases. These dynamics are depicted in Figure 1.

The typical properties of dynamic processes are a non-constant states transition rate (*non-homogeneity*), and the ability to observe the process states at only a finite set of time-points (*point observations*). Additionally, the processes may include highly diverse explanatory variables whose distribution is often difficult to learn. For example, this might include the type of drugs taken by each individual and their viral genome at that time. Due to these properties, a modeling framework for such processes should account for non-homogeneity, deal with point observations, be scalable in the number of components, and provide a robust way to account for observed explanatory variables without modeling their distribution.

The seminal work of (Cox, 1972) laid the foundations for rigorous analysis of the dynamics of non-homogeneous irreversible processes by introducing the *proportional hazard model*. A key point of this model is its focus on modeling the dynamics of a single binary-valued variable *conditioned* on some set of background variables. The proportional hazard framework proposed by Cox turned out to be extremely useful in modeling processes such

as survival after medical procedures, how specific drugs affect a disease, the failure of manufactured components, and many more.

In recent years, several extensions of this model have been proposed (e.g., (Du et al., 2013)). One notable extension of the Cox model is Multi-State models (MSTMs), which model single component processes that can occupy one of a finite number of states at each time point (Putter et al., 2007). MSTMs support non-homogeneity and learning from point observations, and allow us to condition the dynamics on explanatory variables, resolving the difficulties in explicitly modeling covariates.

MSTMs are increasingly being used in medical and epidemiological studies (e.g., (Looker et al., 2013; Walker et al., 2013; Taghipour et al., 2013)). Nevertheless, MSTMs are not naturally suited for analyzing multi-component processes, because they require defining a state space corresponding to the Cartesian product of the individual components, resulting in a representation that is exponential in the number of components.

In this paper we consider modeling the conditional distribution of a non-homogeneous multi-dimensional continuous-time Markov process $\mathbf{Y}(t) = Y_1(t), \dots, Y_n(t)$ given a set of covariates $\mathbf{x} \in R^p$, which we refer to as background covariates. Our goal is to construct a modeling language that is compact, interpretable, and scalable, meaning that it allows learning dependencies of specific components on specific covariates as well as on other components, while allowing efficient inference and learning.

A Continuous-Time Bayesian Network (CTBN) (Nodelman et al., 2002), the continuous-time extension of a dynamic Bayesian network, is a framework that enables the modeling of high-dimensional processes with complex dependencies; these dependencies are expressed via an interpretable network topology. CTBNs naturally deal with missing data, using exact inference for small topologies, and a variety of approximate methods for large topologies. Therefore, the principles that underlie CTBNs can serve as a basis to scale up MSTMs, by introducing structured representation and accompanying mathematical machinery from CTBNs.

In this work, we define StruCTured proportional jUmP Processes (SCUP), a new model combining ideas from the fields of proportional hazard models, MSTMs, and CTBNs. Our key modeling assumption decomposes the dynamics of the process into two elements. The first element is the *effect of time* on the dynamics of each individual component, independently of the others. The second element represent the dependence of the *evolution of each component* on the background covariates, as well as on the state of the other components. This decomposition allows a compact representation of

the combined effect of non-homogeneity, background variables, and interactions among components. We show how this model can be learned from point observations and demonstrate the properties of our approach on synthetic data, as well as on large cohorts of data from diabetes and HIV patients. Our analysis helps identify reliable markers that may predispose diabetes and HIV patients to medical complications. Namely, we find that routine blood tests can serve as a biomarker for an increase in glycated hemoglobin, which is a highly reliable marker for complications among diabetes patients.

2 BACKGROUND

A multi-component continuous-time stochastic process over a discrete state space is a collection of vector-valued random variables $\{\mathbf{Y}(t) = Y_1(t), \dots, Y_n(t) | t \geq 0\}$ where for each component i , $Y_i(t)$ takes on values from a finite set S_i . We say that such a process is *Markovian* if, for all sequences $t_1 \leq t_2 \leq \dots \leq t_k$, it satisfies

$$\begin{aligned} \Pr(\mathbf{Y}(t_k) = \mathbf{y}_k | \mathbf{Y}(t_{k-1}) = \mathbf{y}_{k-1}, \dots, \mathbf{Y}(t_1) = \mathbf{y}_1) \\ = \Pr(\mathbf{Y}(t_k) = \mathbf{y}_k | \mathbf{Y}(t_{k-1}) = \mathbf{y}_{k-1}) \end{aligned}$$

Continuous time Markov processes are completely characterized by the rate function $q(t; \mathbf{y}, \mathbf{y}')$, which determines the instantaneous transition probability between states:

$$\Pr(\mathbf{Y}(t+\Delta t) = \mathbf{y}' | \mathbf{Y}(t) = \mathbf{y}) = 1_{\mathbf{y}=\mathbf{y}'} + q(t; \mathbf{y}, \mathbf{y}')\Delta t + o(\Delta t) \quad (1)$$

where 1 is the indicator function and $o(\Delta t)$ is a function that converges to zero faster than its argument, i.e., $\lim_{\Delta t \downarrow 0} \frac{o(\Delta t)}{\Delta t} = 0$. The rate functions are non-negative for every $\mathbf{y} \neq \mathbf{y}'$. The diagonal elements $q(t; \mathbf{y}, \mathbf{y})$ are the exit rates from state \mathbf{y} at time t that satisfy $q(t; \mathbf{y}, \mathbf{y}) = -\sum_{\mathbf{y}' \neq \mathbf{y}} q(t; \mathbf{y}, \mathbf{y}')$. A Markov jump process is *homogeneous* if the rates do not depend on time, i.e., $q(t; \mathbf{y}, \mathbf{y}') = q_{\mathbf{y}, \mathbf{y}'}$, otherwise it is *non-homogeneous*.

Continuous time Bayesian networks (CTBNs) provide a compact representation for homogeneous Markov processes where only one component can change at a time, and where the instantaneous dynamics of each component i are influenced by a small set of parent components denoted by $pa(i)$. We refer to $pa(i)$ as the *context* of the component i . These assumptions are encoded by setting $q(t; \mathbf{y}, \mathbf{y}') = 0$ when \mathbf{y} and \mathbf{y}' differ by more than one component, and $q(t; \mathbf{y}, \mathbf{y}') = q_{y_i, y'_i | \mathbf{y}_{pa(i)}}$ when they differ in component i , where y_i and $\mathbf{y}_{pa(i)}$ are the states of component i and of the subset $pa(i)$, respectively. This dependency structure is represented by a directed graph \mathbf{G} over the nodes labeled $1 \dots n$, where the parents of node i are $pa(i)$. We note that the graph G need not be a DAG. In recent years, several approximate methods that exploit this structured representation have been developed (Saria et al.,

2007; Cohn et al., 2010; El-Hay et al., 2010; Celikkaya et al., 2011; Rao and Teh, 2011b; Oppen and Sanguinetti, 2007).

3 STRUCTURED PROPORTIONAL JUMP PROCESSES

Consider a system of interacting components with two additional characteristics: (1) The dynamics of each component depends on a set of background variables $\mathbf{x} \in R^P$; and (2) Transition rates are non-homogeneous. This work deals with modeling and learning the interactions between the components as well as the relation between the background variable \mathbf{x} and the dynamics of the system represented by $\mathbf{Y}(t)$. As in regression and conditional models, the distribution of the background covariates \mathbf{x} will not be modeled.

Assuming Markovian dynamics, such systems are characterized by a conditional rate function $q(t; \mathbf{y}, \mathbf{y}' | \mathbf{x})$. To model this rate in a compact manner we first assume that, as in CTBNs, \mathbf{Y} has local dynamics, namely is governed by conditional rate functions for all $\mathbf{y} \neq \mathbf{y}'$:

$$q(t; \mathbf{y}, \mathbf{y}' | \mathbf{x}) \equiv q^i(t; y_i, y'_i | \mathbf{y}_{pa(i)}, \mathbf{x}) \cdot 1_{\{j: y_j \neq y'_j\} = \{i\}}.$$

Next, we need to capture the dependency of these dynamics on both time and covariates. To do this, we decompose the rate into two elements: the dependence on time and the joint dependence on context and background variables.

The effect of the time on the transition is captured by the notion of the *baseline rate*. For each component i and pair of states y_i to y'_i , we denote by $r_{y_i, y'_i}^i(t)$ the non-negative time dependent functions. The effect of the joint state of the covariates \mathbf{x} and the context $\mathbf{y}_{pa(i)}$ is mediated through a set of weight vectors $\mathbf{w}_{y_i, y'_i}^i \in R^N$. Combining these elements, we define the conditional transition of the SCUP model:

$$q^i(t; y_i, y'_i | \mathbf{y}_{pa(i)}, \mathbf{x}) \equiv r_{y_i, y'_i}^i(t) \cdot \exp\{\mathbf{w}_{y_i, y'_i}^i \cdot \phi^i(\mathbf{x}, \mathbf{y}_{pa(i)})\}, \quad (2)$$

where $\phi^i(\mathbf{x}, \mathbf{y}_{pa(i)})$ is a mapping of the covariates and parent states into an N -dimensional feature vector (where in general N could depend on i). This representation does not explicitly specify the dependency structure of the components on \mathbf{x} because it does not have a significant effect on the inference computational complexity, as shown in Section 4. Note that the time-dependent effect is common to the entire population, meaning that it does not depend on the covariates \mathbf{x} and $\mathbf{y}_{pa(i)}$. On the other hand, the covariates, as well as the parent components, modulate the transition rate between states y_i and y'_i through the second element, independently of time.

To gain some insight into the assumptions encoded in this model, we consider three examples. First, we note that

by setting the baseline rates to a constant value, removing the background variables, and setting ϕ^i to be a vector of indicators of the parents' joint state, we obtain a CTBN.

The second example is the *Cox proportional hazard model* (Cox, 1972). This model has a single binary outcome Y where $Y = 1$ represents a base state and $Y = 0$ represents a terminal *failure* state such as death. The rates in such a system are:

$$q(t; 1, 0 | \mathbf{x}) \equiv r_0(t) e^{\mathbf{w} \cdot \mathbf{x}} \quad \text{and} \quad q(t; 0, 1 | \mathbf{x}) \equiv 0, \quad (3)$$

where $r_0(t)$ is the baseline rate. In this model $q(t; 1, 0 | \mathbf{x})$ is called the *hazard function* and the probability of surviving for a time greater than t is

$$\Pr(Y(t) = 1 | \mathbf{x}, Y(0) = 1) = e^{-\int_0^t q(s; 1, 0 | \mathbf{x}) ds}.$$

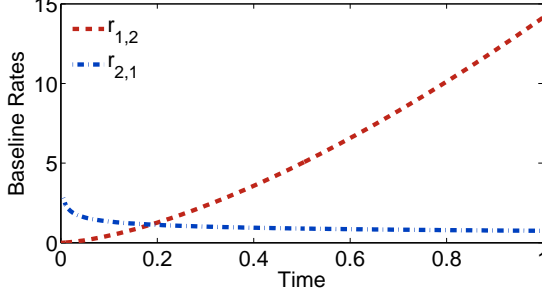
In case the baseline is constant, the survival time distribution is exponential. A monomial baseline, $r_0(t) = \lambda k (\lambda t)^{k-1}$, gives a Weibull distribution. A common approach is to model the baseline in a non-parametric manner (see the seminal work of (Kaplan and Meier, 1958)).

The Cox model encapsulates an assumption that the failure rate proportion for two individuals with attributes \mathbf{x}_1 and \mathbf{x}_2 is time invariant as $q(t; 1, 0 | \mathbf{x}_1) / q(t; 1, 0 | \mathbf{x}_2) = e^{\mathbf{w} \cdot (\mathbf{x}_1 - \mathbf{x}_2)}$. This approach is generalized in *multi-state-models* (Putter et al., 2007), which involve a single component and define $q(y, y' | \mathbf{x}; t) = r_{y, y'}(t) e^{\mathbf{w}_{y, y'} \cdot \mathbf{x}}$, resulting in a proportion of $e^{\mathbf{w}_{y, y'} \cdot (\mathbf{x}_1 - \mathbf{x}_2)}$.

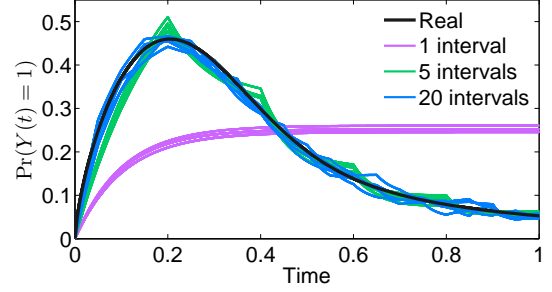
The proportionality assumption in SCUP is conditional, that is, if we fix $\mathbf{Y}_{pa(i)}(t) = \mathbf{y}_{pa(i)}$ the proportion between conditional rates is $\exp\{\mathbf{w}_{y_i, y'_i}^i \cdot (\phi^i(\mathbf{x}_1, \mathbf{y}_{pa(i)}) - \phi^i(\mathbf{x}_2, \mathbf{y}_{pa(i)}))\}$ for all t . However, the proportion of the actual marginal rate of moving from y_i to y'_i is time dependent because it is marginalized with time dependent weights, $\Pr(\mathbf{Y}_{pa(i)}(t) | \mathbf{x})$. A time invariance property also holds for proportions between transition rates that are conditioned on two different parent assignments for a fixed \mathbf{x} .

The third example deals with an HIV patient model, as shown in Figure 1. The proposed model contains components corresponding to the viral load (VL), CD4 concentration, the status of a certain disease of interest, and an absorbing survival component. The model topology encodes the assumption that the VL and CD4 components affect each other directly, whereas the effect of VL on survival is mediated through CD4 and the disease.

As a concrete example, the disease state space can be {"none", "mild", "severe"}, with the possible transitions "none" \leftrightarrow "mild", and "mild" \leftrightarrow "severe", and the CD4 state space can be {"high", "low"}. The CD4 \rightarrow disease arc encodes a parameter for each combination of the CD4 level and one of the four disease transitions. Notably, the



(a) True baseline rates



(b) Approximation of predictive probability

Figure 2: A Piecewise linear approximation for a non-homogeneous process.

ratio between the transition rates given CD4=“high” and given CD4=“low” is time independent, and determined solely according to the mapping $\phi^{disease}(\text{CD4})$ and the parameters $w^{disease}$ for each transition.

3.1 REPRESENTATION OF BASELINE RATES

Time dependent baseline rates can either be represented non-parametrically as in the classical Cox model, or assume a parametric representation. Examples include Weibull hazard function $r(t) = \lambda k(\lambda t)^{k-1}$, log-logistic hazard, $r(t) = \frac{\lambda k t^{k-1}}{1 + \lambda t^k}$ and more. In this work we will adopt a piecewise constant representation, which can approximate well behaved processes with a high degree of accuracy.

To characterize such processes, we consider single-component models with a time-dependent state $Y(t)$ (every model can be represented as a single-component model whose state space is the Cartesian product of the components state spaces). Denote by $\mathbf{P}^{\mathbf{Q}}(s, t)$ the transition matrix whose y, y' entry is $\Pr(Y(t) = y | Y(s) = y')$, and by $\mu_y^{\mathbf{Q}}(t) \equiv \Pr^{\mathbf{Q}}(Y(t) = y)$ the time-dependent marginal probability. We say that a matrix \mathbf{P} is embeddable if there exists a matrix \mathbf{A} such that $\mathbf{P} = e^{\mathbf{A}}$. Let $\tau_0 < \tau_1 < \dots < \tau_K$ be an ordered set of time points, and suppose that $\mathbf{P}^{\mathbf{Q}}(\tau_{k-1}, \tau_k)$ is embeddable for every $k = 1, \dots, K$. From the Markov property, it follows that there exists a piecewise constant rate matrix function $\hat{\mathbf{Q}}(t) = \mathbf{Q}_k, \forall \tau_{k-1} \leq t < \tau_k$ that satisfies $\mu_y^{\mathbf{Q}}(\tau_k) = \mu_y^{\hat{\mathbf{Q}}}(\tau_k)$. Moreover, the following lemma bounds the error for rate matrices with bounded transition rates:

Lemma 3.1 : *Let $Y(t)$ be a non-homogeneous process with bounded transition rates $\mathbf{Q}(t)$ and an embeddable rate matrix. Denote $\rho_k = \max_{y, \tau_{k-1} \leq t < \tau_k} |q_{y,y}(t)|$, $\hat{\rho}_k = \max_y |\hat{q}_{y,y}|$. Then, for all y and $\tau_{k-1} \leq t \leq \tau_k$, $|\mu_y^{\mathbf{Q}}(t) - \mu_y^{\hat{\mathbf{Q}}}(t)| < (\rho + \hat{\rho}) \cdot (\tau_k - \tau_{k-1})$.*

This lemma, proven in the appendix, suggests that the

number of intervals required to bound the bias by ϵ scales inversely linear with $1/\epsilon$. We note that tight approximations exist in the case of non-embeddable processes (Davies, 2010).

As an example, consider a two state model with time dependent baseline rates depicted in Figure 2a. This process induces a non-monotone marginal probability $\mu_1^{\mathbf{Q}}(t)$ given an initial condition $Y(0) = 2$, as shown by the smooth black line in Figure 2b. The initial rise follows from the relation $r_{2,1}(t) > r_{1,2}(t)$, and the subsequent decline from the opposite relation. The colored lines show estimated probabilities given by piecewise constant models with 1, 5 and 20 intervals of constant rates that were trained on 1000 simulated trajectories.

4 LEARNING

Generally, training data may include a mixture of point observations on some components and full (complete) trajectories of others. For example CD4 and viral load are point observations measured periodically, whereas time of death is usually exactly recorded resulting in a continuous observation on survival. To learn from such data, we will adapt the approach taken for CTBNs, which handles unseen state trajectories between observations as missing data and employs Expectation Maximization (EM). The first step is to derive the likelihood of the model given complete trajectories.

4.1 LIKELIHOOD FUNCTION

A fully observed trajectory is represented using the sequence t_0, \dots, t_M and states y_0, \dots, y_{M-1} such that $Y(t) = y_k$ for $t \in [t_k, t_{k+1})$. We denote such a trajectory by $y_{[0, t_M]}$. The likelihood of a non-homogeneous process

with a set of rates $\mathcal{M} = \{q(t; y, y')\}_{y, y'}$ is

$$l(\mathcal{M}|y_{[0, t_M]}) = \exp \left\{ \int_{t_{M-1}}^{t_M} q(t; y_{M-1}, y_{M-1}) dt \right\} \quad (4)$$

$$\cdot \prod_{k=0}^{M-2} \left[\exp \left\{ \int_{t_k}^{t_{k+1}} q(t; y_k, y_k) dt \right\} q(t_{k+1}; y_k, y_{k+1}) \right]$$

where $q(t; \mathbf{y}, \mathbf{y}) = -\sum_{y' \neq y} q(t; y, y')$.

Let \mathcal{D} be a data set that includes pairs of trajectories $y^{[c]}$ and covariates \mathbf{x}_c , where $c = 1, \dots, N_{\text{sequences}}$ and denote by $ll(\mathcal{M}|\mathcal{D})$ be the log-likelihood.

The log-likelihood is unbounded if there are no constraints on the baseline rate functions. Consider for example the survival model described in Equation 3, and suppose that no background variable is involved. In this case,

$$ll(r_0(t)|\mathcal{D}) = \sum_c \left[-\int_0^{t_c} r_0(t) dt + \log r_0(t_c) \right].$$

One can construct a series of baseline rates such that this term approaches infinity as $r_0(t) \sum_c a_c \delta(t - t_c)$, implying that a naive maximum likelihood procedure tends to overfit $r_0(t)$ to a function that imposes transitions at the observed times if no constraints are put in place. An alternative approach for non-parametric estimation of a possibly arbitrary baseline is to use partial-likelihood (Cox, 1972, 1975). However, this direction does not generalize naturally to partially observed data. Two possible approaches for placing constraints use either a restricted parametric form or regularized baseline.

4.2 PARTIALLY OBSERVED DATA

To deal with partially observed data, we perform an EM procedure. On each iteration we compute the expected log-likelihood of a new model with respect to the posterior distribution of the current model \mathcal{M}_0 . The posterior distribution of a Markov process \mathcal{M}_0 given a sequence σ_c is characterized by a set of time-dependent functions (Cohn et al., 2010)

$$\mu_y(t|c) = \Pr(Y(t) = y|c, \mathcal{M}_0)$$

$$\gamma_{y, y'}(t|c) = \lim_{\Delta t \rightarrow 0} \frac{\Pr(Y(t) = y, Y(t + \Delta t) = y'|c, \mathcal{M}_0)}{\Delta t}$$

$\mu_y(t|c)$ is the singleton probability that the process is in state y at time t . $\gamma_{y, y'}(t|c)$ is the *intensity* of the pairwise probability of being in state y and then moving to y' at time t .

Using this characterization, taking the expectation on the log of the likelihood function in Equation 4 and plugging in the decomposition of the conditional intensities depicted

in Equation 2, gives the expected log-likelihood of a multi-component model:

$$E_{\mathcal{M}_0}[ll(\mathcal{M}, \mathcal{D})] = \sum_c \sum_{y_i, \mathbf{y}_{pa(i)}} \sum_{y'} \quad (5)$$

$$\left[-\exp\{\mathbf{w}_{y_i, y'_i}^i \cdot \phi^i(\mathbf{x}_c, \mathbf{y}_{pa(i)})\} \int_t \mu_{y_i, \mathbf{y}_{pa(i)}} r_{y_i, y'_i}^i dt \right.$$

$$\left. + \int_t \gamma_{y_i, y'_i | \mathbf{y}_{pa(i)}} \left(\log r_{y_i, y'_i}^i + \mathbf{w}_{y_i, y'_i}^i \cdot \phi^i(\mathbf{x}_c, \mathbf{y}_{pa(i)}) \right) dt \right]$$

where we omit t and c from μ , γ and r , $\mu_{y_i, \mathbf{y}_{pa(i)}}$ is the marginalization of the posterior distribution to the subset of components $i, pa(i)$, and similarly $\gamma_{y_i, y'_i | \mathbf{y}_{pa(i)}} = \sum_{\{\hat{\mathbf{y}}_i = y_i, \hat{\mathbf{y}}_{pa(i)} = \mathbf{y}_{pa(i)}\}} \gamma_{\hat{\mathbf{y}}, [\hat{\mathbf{y}} \setminus i, y'_i]}$ is a marginalization of pairwise probability intensities. Additional details are given in the Appendix. An exact computation of these functions and their integrals is feasible for systems with a small number of components. Otherwise, a variety of approximate methods are available (Saria et al., 2007; Cohn et al., 2010; El-Hay et al., 2010; Celikkaya et al., 2011; Rao and Teh, 2011b; Oppen and Sanguinetti, 2007).

4.3 OPTIMIZATION

The gradient of the log-likelihood with respect to \mathbf{w} is:

$$\frac{\partial E_{\mathcal{M}_0}[ll(\mathcal{M}, \mathcal{D})]}{\partial \mathbf{w}_{y_i, y'_i}^i} =$$

$$\sum_c \sum_{\mathbf{y}_{pa(i)}} \phi^i(\mathbf{x}_c, \mathbf{y}_{pa(i)}) (M_{\mathbf{y}_{pa(i)}}^c - MP_{\mathbf{y}_{pa(i)}}^c)$$

where the first term $M_{\mathbf{y}_{pa(i)}}^c = \int_t \gamma_{y_i, y'_i | \mathbf{y}_{pa(i)}} dt$ is the expected number of transitions from y_i to y'_i given the state of the parents $\mathbf{y}_{pa(i)}$, M_0 and the evidence in sequence c (see (Cohn et al., 2010)). The second term

$$MP_{\mathbf{y}_{pa(i)}}^c = \exp\{\mathbf{w}_{y_i, y'_i}^i \cdot \phi^i(\mathbf{x}_c, \mathbf{y}_{pa(i)})\} \int_t \mu_{y_i, \mathbf{y}_{pa(i)}}^i r_{y_i, y'_i}^i dt$$

is the integral of the probability of being in state $[y_i, \mathbf{y}_{pa(i)}]$, multiplied by the transition rate. Hence, this term can be interpreted as the expected number of *potential transitions*. The gradient weighs the feature vectors $\phi^i(\mathbf{x}_c, \mathbf{y}_{pa(i)})$ using the difference between the expected number of actual and potential transitions.

Optimization of the baseline that assumes a parametric form $r_{y_i, y'_i}^i(t) = r_{y_i, y'_i}^i(t; \theta)$ involves computation of its gradient with respect to the parameters θ

$$\frac{\partial E_{\mathcal{M}_0}[ll(\mathcal{M}, \mathcal{D})]}{\partial \theta} =$$

$$\sum_c \sum_{\mathbf{y}_{pa(i)}} \int_t \left[-\exp\{\mathbf{w}_{y_i, y'_i}^i \cdot \phi^i(\mathbf{x}_c, \mathbf{y}_{pa(i)})\} \mu_{y_i, \mathbf{y}_{pa(i)}} \right.$$

$$\left. + \frac{\gamma_{y_i, y'_i | \mathbf{y}_{pa(i)}}}{r_{y_i, y'_i}^i} \right] \frac{\partial r_{y_i, y'_i}^i}{\partial \theta} dt.$$

In the simplest case, if the baseline is constant or piecewise constant, the stationary point solution has a closed form.

A maximum likelihood estimator can be found using an EM procedure iterating between expectation and maximization steps. Expectation steps compute the functions that represent the posterior distribution, μ and γ . Maximization steps involve optimizing the covariate and cross component influence weights \mathbf{w}_{y_i, y'_i}^i using standard optimization methods and the gradient derived in Equation 6, as well as optimizing the baseline rates using the gradient in Equation 6. While the overall target function is not convex, the optimization of \mathbf{w}_{y_i, y'_i}^i is a convex problem given fixed baselines and posterior distributions, and so is the case for many choices of the baseline rates.

5 EXPERIMENTAL RESULTS

5.1 LEARNING EVALUATION

Our initial experiments test the validity of SCUP. To this end, we created synthetic SCUP data sets. We then trained SCUP using these data sets, and compared the similarity of the learned models with the actual ones.

The topology for all data generating models was similar to the HIV disease topology (Figure 1) with the exclusion of the survival state. All models included a single randomly drawn binary covariate. The baseline rates followed a Weibull rate, with a shape parameter $\kappa = 2$ and a scale parameter drawn from an inverse Gamma distribution (the Weibull distribution conjugate prior), with shape and scale parameters both equal to 2. For each component with parents y_1, y_2 , and a covariate x , we used the feature mapping $\phi(x, y_1, y_2) = (x, \mathbf{1}_{y_1=2}, \mathbf{1}_{y_2=2})$, with feature coefficients drawn from $\mathcal{N}(0, 1)$.

We evaluated learning performance as a function of dataset size and sampling rate. During the training, we divided the time interval $[0, 1]$ into 5 equally sized intervals, and learned piecewise-constant baseline rates for each one. We considered both fully observed data and point observations, with observation times for each trajectory drawn uniformly from $[0, 1]$. All trajectories were observed at times $t = 0$ and $t = 1$. Our evaluation compared the similarity of the learned models to the true generating models through the root mean square error (RMSE) of the learned coefficients. We also compared the integral of each baseline rate across the time interval $[0, 1]$, to its true value. The baseline integral was used because it does not depend on parametric form, and because it is used in inference and learning tasks, rather than the baseline itself. Figure 3 shows that learning accuracy increases with sample size and sampling rate, as expected. As a further measure of validity, we verified that $\log(\text{RMSE})$ decreases linearly with \log dataset size, with slopes close to -0.5, indicating consistency (data not shown).

5.2 THE EFFECT OF NON-HOMOGENEITY

To test the effect of non-homogeneity, we generated data from homogeneous and non-homogeneous SCUP models. We then trained the models with different levels of non-homogeneity on the generated datasets, and evaluated learning performance. The datasets were generated from two SCUP architectures similar to those described in the previous section, with the exception that the first architecture used a homogeneous constant baseline rate for all transitions, whereas the second one used baseline rates as previously described. The baseline rates for the first architecture were generated from a Gamma distribution, with scale and shape parameters equal to 1.0. We generated five models from each architecture, and generated a dataset of 500 trajectories using each model. Every trajectory was observed at times $t = 0$ and $t = 1$, and at three other uniformly drawn time-points. We trained SCUP models with increasing numbers of piecewise-constant baseline rate. Notably, models with one baseline rate are equivalent to CTBNs. We evaluated the learning performance via a five-fold cross validation of out of sample (OOS) likelihood.

The results, shown in Figure 4, demonstrate that homogeneous models cannot capture complex dynamics that change over time. Increasing the number of baseline rates leads to greater flexibility on the one hand, but to the risk of overfitting on the other.

5.3 COMPARISON WITH OTHER METHODS

To assess the relative performance of SCUP, we compared it to two competing methods, which can both be derived as special instances of SCUP: A factored model and a multi-state model. The factored model (FM) is a SCUP model with several independent components. There are no arcs between components, and thus transition probabilities are affected only by covariates and baseline rates. The multi-state model (MSM) follows the implementation of a package called MSM (Jackson, 2011). It can be viewed as a SCUP model with a single component, whose state space is the Cartesian product of the SCUP components state spaces. We verified empirically that our implementation yields the same results as MSM on a wide variety of scenarios. SCUP can be seen as an intermediate method between these two extremes, balancing between compactness and expressiveness. Notably, all three methods fully support non-homogeneous dynamics.

We generated five models for each of the three architectures, each having a single binary covariate, with Weibull baseline rates and randomly drawn coefficients, as described in the previous section. The SCUP models were generated and used as described in the previous sections. The FM models contained three binary components, and the MSM models contained a single eight-state component,

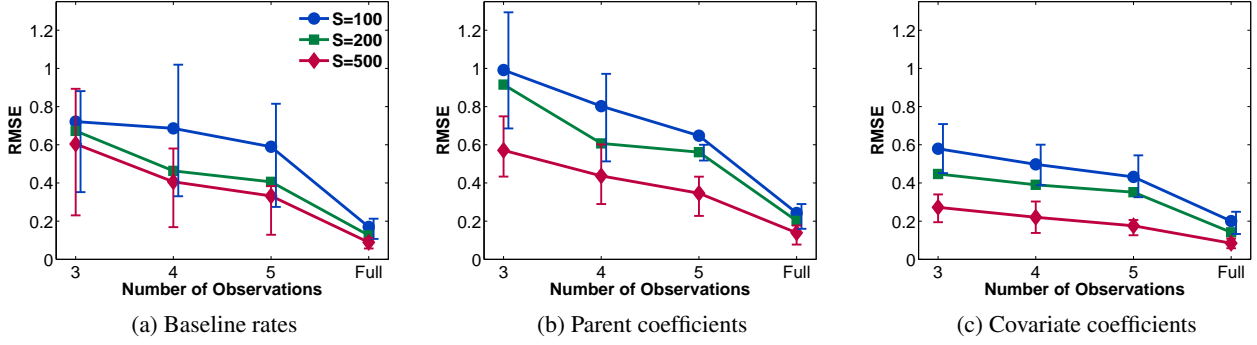


Figure 3: Root mean square error of estimated parameters for various sampling rates, and the 75% confidence intervals (confidence intervals for $S=200$ are omitted for clarity).

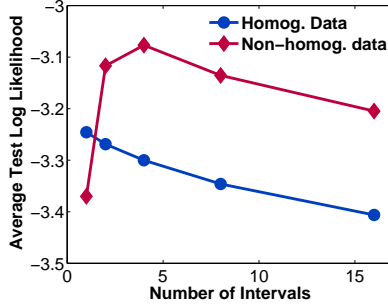


Figure 4: Out of sample likelihood for models trained with increasing number of piecewise-constant baseline rates.

Table 1: The number of parameters learned by FM, SCUP, and MSM. The number of piecewise-constant baseline rates is denoted by t .

	FM	SCUP	MSM
Cov. coefficients	6	6	56
Parents coefficients	0	12	0
Baseline rates	$6t$	$6t$	$56t$
Total number	$6+6t$	$18+6t$	$56+56t$

with one state corresponding to each assignment of the components' states in the SCUP model. Both the MSM and FM models used the feature mapping $\phi(x) = x$.

We generated datasets of 1,000 trajectories using each of the 15 models. We then examined how well a model from each architecture can be trained on each dataset, via a three-fold cross validation of OOS likelihood. The trajectories were observed as described in the previous section. All trained models used five piecewise constant baseline rates. The number of parameters for the three models is shown in Table 1, demonstrating that SCUP bridges between the two extremes.

Figure 5 demonstrates that SCUP is more flexible than the other two methods, allowing it to represent data generated

by different architectures, while retaining compactness. MSM exhibits poor learning capabilities for smaller datasets; this holds true even for data created by a model with the same architecture, demonstrating overfitting due to model complexity. The factored model does not suffer from overfitting, but has limited expressiveness, and thus cannot capture mutual influences between components.

5.4 ANALYSIS HIV DATA

We evaluated the performance of SCUP by analyzing real data from a data set containing lab measurements of HIV patients who took medication on a regular basis, previously described in (Rosen-Zvi et al., 2008). We defined models with two components corresponding to the two main measures of HIV severity, viral load (VL) and CD4 lymphocytes concentration, as well as a continuously observed binary absorbing component, representing survival. The resulting model is similar to the one described in Figure 1, with the omission of the disease component, and the addition of a $VL \rightarrow \text{survival}$ arc, which was added to obtain a fully connected topology. Following previous works, the CD4 level was dichotomized to have 2 states, using a threshold of 200 (D'Amico et al., 2011). The VL level was also dichotomized to have 2 states, using a threshold of 500, as previously done in analyses of this data (Rosen-Zvi et al., 2008).

For the analysis, we randomly selected 2000 patients whose VL and CD4 levels were both observed at each observation point. The resulting dataset contained 5.14 observations per patient on average (standard deviation 3.37). For every patient, we included covariates corresponding to age, sex, and whether the patient had undertaken a different therapy in the past. Feature mappings consisted of a concatenation of the covariates, with a binary 0/1 feature for each parent component. The initial time $t=0$ was set as the therapy start time. For patients who underwent several successive therapies, only observations taken during the period of the first one were included in the analysis. All patients had

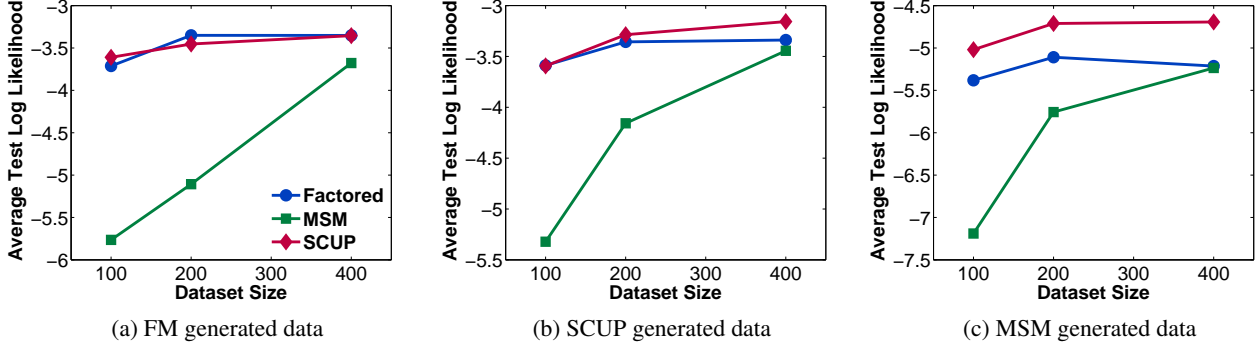


Figure 5: Test likelihoods of data learned by different models.

an observation at time $t=0$, using the closest measurement within a month from the therapy start date.

We computed the average OOS log likelihood obtained via a five-fold cross validation, with increasing numbers of piecewise constant baseline rates. The results, shown in Figure 6, clearly demonstrate the powerful effect of non-homogeneity, and the importance of modeling it correctly. MSM has an advantage when using a small number of baseline rates, owing to its richer model, which can capture richer interaction patterns between the system components. However, SCUP steadily improves as the number of baseline rates increases, until it eventually surpasses MSM. This increase indicates the presence of strong non-homogeneous dynamics. MSM can also capture non-homogeneous dynamics, but is hindered by its large number of parameters. The FM model exhibits weaker performance than the other methods for every number of baseline rates tested. This is due to the fact that it cannot capture the dynamics stemming from mutual influences between the system components. The decrease in OOS likelihood for FM when using 16 baseline rates may stem from overfitting, which occurs because it is trying to incorrectly capture mutual influences between the system components via baseline rates.

5.5 ANALYSIS OF DATA FROM DIABETES PATIENTS

We evaluated SCUP on a large cohort of diabetes patients, previously described by (Neuvirth et al., 2011). Following (Neuvirth et al., 2011), we define the main outcome of interest as the glycated hemoglobin (HbA1c) blood test, which is a reliable indicator of diabetes severity status. A higher HbA1c indicates increased risk of developing complications.

Our goal was to learn the interaction patterns between the HbA1c level and other potential diabetes biomarkers commonly measured in routine blood tests. The ability to predict HbA1c levels from routine blood tests can improve early detection of the disease progression. To this end,

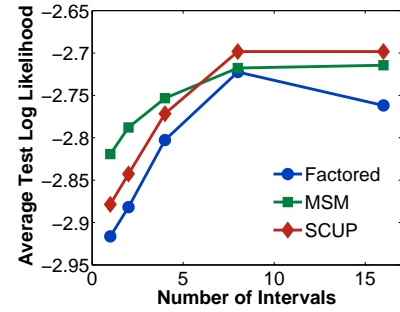


Figure 6: Performance of SCUP, FM, and MSM on the HIV dataset.

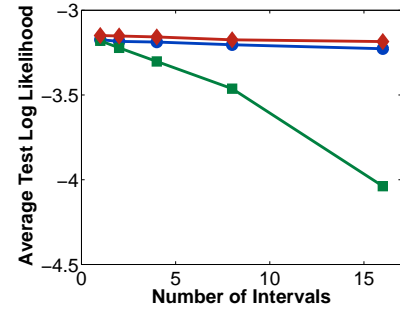


Figure 7: Performance of SCUP, FM, and MSM on the diabetes dataset.

we defined a SCUP model with binary components for HbA1c, low-density lipoprotein (LDL), and triglycerides levels. The two states of each component correspond to normal and abnormal clinical status, with the thresholds for HbA1c, LDL, and triglycerides set to 7, 130 and 200, respectively. We used a fully connected topology, and included the age and sex of each patient as covariates.

For the analysis, we randomly chose 1,000 patients with non-missing values for the components of interest at every observation point. Every patient had 3.25 observations on average (standard deviation 1.52). Feature mappings consisted of a concatenation of the covariates, with a

binary 0/1 feature for each parent component. The time $t = 0$ for each patient was determined according to the first observation time.

We computed the average OOS likelihood obtained via a five-fold cross validation, with increasing numbers of intervals. The results, shown in Figure 7, demonstrate that SCUP can scale to rich models without overfitting. The factored model, although scalable, does not capture the interactions between components, leading to weaker prediction power. The MSM model tends to suffer from overfitting due to its complexity. The lack of increase in OOS likelihood for increased numbers of intervals indicates that the components tend to follow homogeneous dynamics in this dataset. Nevertheless, SCUP does not overfit when trained with a large number of intervals, indicating its robustness to the type of underlying dynamics in the data.

To further investigate the different methods, we examined the coefficients describing mutual influence between the system components; these were learned across the different folds. We examined the models that assumed one piecewise-constant interval, as they had the best fit for this data. For every pair of components, we computed the coefficient describing the influence of one on a transition of the other. For MSM, we averaged the two corresponding coefficients over the two possible states of the third component. The results, shown in Table 2, demonstrate that SCUP models learned across the different folds are more consistent with each other, leading to substantially smaller variance.

The results demonstrate rich interaction patterns across the components. For example, increased triglycerides levels are associated with an increase in HbA1C, whereas increased HbA1C is associated with stabilization of the triglycerides levels via a reduction of their transition rate. Such observations cannot be performed directly in FM nor MSM, due to their lack of modular structure.

6 DISCUSSION

We proposed a proportional modeling scheme for non-homogeneous multi-component processes, by combining factorizations of CTBNs with a decomposition dating back to proportional hazard models. The key modeling assumption is a decomposition of the process into a time-dependent non-homogeneous component that does not depend on the model topology, and a time-independent component that depends on the model topology and additional features. This is a natural extension of classic hazard models, which can be considered as special SCUP instances with no underlying topology. This decomposition leads to compact models that can capture complex dynamics, as well as an efficient learning scheme, and easily interpretable results.

Table 2: The average coefficients of parent influence on increase (\uparrow) and decrease (\downarrow) learned in the diabetes dataset, and the minimum and maximum values obtained across the five folds.

	SCUP	MSM
LDL \rightarrow A1C \uparrow	.17 (.08, .25)	-.24 (-.73, .27)
Trig. \rightarrow A1C \uparrow	.31 (.09, .45)	.12 (-.65, .62)
LDL \rightarrow A1C \downarrow	.09 (.04, .16)	-.20 (-.50, .06)
Trig. \rightarrow A1C \downarrow	-.17 (-.23, .02)	-.22 (-.42, .22)
A1C \rightarrow LDL \uparrow	.34 (.20, .45)	1.15 (.86, 1.31)
Trig. \rightarrow LDL \uparrow	.57 (.33, .79)	.68 (.31, 1.14)
A1C \rightarrow LDL \downarrow	-.04 (-.23, .14)	-.18 (-.98, .33)
Trig. \rightarrow LDL \downarrow	-.38 (-.49, -.25)	.67 (.16, 1.46)
A1C \rightarrow Trig. \uparrow	-.28 (-.49, -.09)	-.51 (-.90, -.37)
LDL \rightarrow Trig. \uparrow	.82 (.67, .92)	-.54 (-1.13, .25)
A1C \rightarrow Trig. \downarrow	-.59 (-.71, -.50)	-1.24 (-1.82, -.89)
LDL \rightarrow Trig. \downarrow	.63 (.40, .82)	-.72 (-1.79, .09)

Our theoretical and empirical results demonstrate that non-homogeneous dynamics can be captured accurately using a piecewise homogeneous approximation. It would be interesting to compare this baseline rates representation to parametric forms. Learning such models is straightforward and can be performed by plugging in the partial derivative of a specific parametric form to the gradient in Equation 6.

Baseline rates can be regularized via spline approximations (Commenges, 2002; Joly et al., 2009; Farewell and Tom, 2012) or Gaussian process priors (Rao and Teh, 2011a). Splines can also be naturally adapted to regularize piecewise constant rates. This can be done by bounding the difference between rates in adjacent time intervals, or the rate of change of this difference, which is analogous to bounding the first and second derivative, respectively. Regularization of other model parameters, such as the covariate or parents coefficients, can potentially be handled using standard regularization methods such as elastic nets, as recently proposed for Cox regression (Simon et al., 2011).

In this work we studied moderately sized systems. Adapting approximate inference methods developed for CTBNs that support non-homogeneity, such as (Rao and Teh, 2011b) or (El-Hay et al., 2010), could scale up this framework to arbitrarily large systems.

Acknowledgments

We would like to thank our anonymous reviewers for useful comments, Hani Neuvirth-Telem for helping with processing of the diabetes data, the EuResist Network study group for providing the HIV data, Gal Elidan for insightful suggestions on earlier versions of the manuscript, and Amir Globerson for helpful discussions.

References

- E. B. Celikkaya, C. R. Shelton, and W. Lam. Factored filtering of continuous-time systems. In *UAI*, 2011.
- I. Cohn, T. El-Hay, N. Friedman, and R. Kupferman. Mean field variational approximation for continuous-time Bayesian networks. *Journal of Machine Learning Research*, 11:2745–2783, 2010.
- D. Commenges. Inference for multi-state models from interval-censored data. *Stat Methods Med Res*, 11(2):167–182, Apr 2002.
- D. R. Cox. Regression models and life-tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 187–220, 1972.
- D. R. Cox. Partial likelihood. *Biometrika*, 62(2):269–276, 1975.
- G. D’Amico, G. Di Biase, J. Janssen, and R. Manca. HIV evolution: a quantification of the effects due to age and to medical progress. *Informatica*, 22(1):27–42, 2011.
- E. B. Davies. Embeddable Markov matrices. *Electronic Journal of Probability*, 15:1474–1486, 2010.
- N. Du, L. Song, M. Gomez-Rodriguez, and H. Zha. Scalable influence estimation in continuous-time diffusion networks. In *NIPS*, pages 3147–3155, 2013.
- T. El-Hay, I. Cohn, N. Friedman, and R. Kupferman. Continuous-time belief propagation. In *ICML*, pages 343–350, 2010.
- V. T. Farewell and B. D. Tom. The versatility of multi-state models for the analysis of longitudinal data with unobservable features. *Lifetime Data Anal*, Dec 2012.
- C. H. Jackson. Multi-state models for panel data: the MSM package for R. *Journal of Statistical Software*, 38(8):1–29, 2011.
- P. Joly, C. Durand, C. Helmer, and D. Commenges. Estimating life expectancy of demented and institutionalized subjects from interval-censored observations of a multi-state model. *Stat Model*, 9(4):345–360, 2009.
- E. L. Kaplan and P. Meier. Nonparametric estimation from incomplete observations. *Journal of the American statistical association*, 53(282):457–481, 1958.
- H. C. Looker, S. O. Nyangoma, D. T. Cromie, J. A. Olson, et al. Predicted impact of extending the screening interval for diabetic retinopathy: the Scottish Diabetic Retinopathy Screening programme. *Diabetologia*, May 2013.
- H. Neuvirth, M. Ozery-Flato, J. Hu, J. Laserson, et al. Toward personalized care management of patients at risk: the diabetes case study. In *KDD*, pages 395–403. ACM, 2011.
- U. Nodelman, C.R. Shelton, and D. Koller. Continuous time Bayesian networks. pages 378–387, 2002.
- M. Opper and G. Sanguinetti. Variational inference for Markov jump processes. In *NIPS*, 2007.
- H. Putter, M. Fiocco, and R. B. Geskus. Tutorial in biostatistics: competing risks and multi-state models. *Stat Med*, 26(11):2389–2430, May 2007.
- V. Rao and Y. W. Teh. Gaussian process modulated renewal processes. In *NIPS*, pages 2474–2482, 2011a.
- V. Rao and Y. W. Teh. Fast MCMC sampling for Markov jump processes and continuous time Bayesian networks. In *UAI*, pages 619–626, 2011b.
- M. Rosen-Zvi, A. Altmann, M. Prosperi, et al. Selecting anti-HIV therapies based on a variety of genomic and clinical factors. *Bioinformatics*, 24(13):399–406, Jul 2008.
- S. Saria, U. Nodelman, and D. Koller. Reasoning at the right time granularity. In *UAI*, pages 326–334, 2007.
- N. Simon, J. H. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for Cox’s proportional hazards model via coordinate descent. *Journal of Statistical Software*, 39(5):1–13, 3 2011. ISSN 1548-7660.
- S. Taghipour, D. Banjevic, A. B. Miller, N. Montgomery, A. K. Jardine, and B. J. Harvey. Parameter estimates for invasive breast cancer progression in the Canadian National Breast Screening Study. *Br. J. Cancer*, 108(3):542–548, Feb 2013.
- A. Walker, S. Doyle, J. Posnett, and M. Hunjan. Cost-effectiveness of single-dose tamsulosin and dutasteride combination therapy compared with tamsulosin monotherapy in patients with benign prostatic hyperplasia in the UK. *BJU Int.*, Jan 2013.

Electing the Most Probable Without Eliminating the Irrational: Voting Over Intransitive Domains

Edith Elkind

University of Oxford, UK
elkind@cs.ox.ac.uk

Nisarg Shah

Carnegie Mellon University, USA
nkshah@cs.cmu.edu

Abstract

Picking the best alternative in a given set is a well-studied problem at the core of social choice theory. In some applications, one can assume that there is an objectively correct way to compare the alternatives, which, however, cannot be observed directly, and individuals' preferences over the alternatives (votes) are noisy estimates of this ground truth. The goal of voting in this case is to estimate the ground truth from the votes. In this paradigm, it is usually assumed that the ground truth is a ranking of the alternatives by their true quality. However, sometimes alternatives are compared using not one but multiple quality parameters, which may result in cycles in the ground truth as well as in the preferences of the individuals. Motivated by this, we provide a formal model of voting with possibly intransitive ground truth and preferences, and investigate the maximum likelihood approach for picking the best alternative in this case. We show that the resulting framework leads to polynomial-time algorithms, and also approximates the corresponding \mathcal{NP} -hard problems in the classic framework.

1 INTRODUCTION

Typically, voting rules are viewed as vehicles for aggregating subjective preferences of individuals into a consensus or societal preference. However, another paradigm of voting theory, which dates back to Marquis de Condorcet [11], became increasingly popular in recent years, motivated in part by its relevance to the design of crowdsourcing platforms and human computation systems [14]. Condorcet suggested that votes cast by the individuals should be viewed as noisy estimates of an underlying objective ground truth—a ranking of the available alternatives by their true quality, and the aim of voting should be to aggregate the votes in order to uncover the ground truth

and thereby pick the best alternative. He proposed a simple approach for modeling the noise present in individuals' votes, which is known today as Mallows' model [18]. In this model, every voter compares each pair of alternatives independently, and orders them correctly (as in the ground truth) with a fixed probability $p > 1/2$, and incorrectly with probability $1-p$. If the generated vote contains cycles, it is discarded and the process restarts, continuing until the pairwise comparisons form a total order over the alternatives. While this model is somewhat unrealistic [19], it is widely used, in part because it provides control of the level of noise in the votes through a single parameter.

However, in many applications, alternatives are compared using not one, but multiple quality parameters [21, 28]. Under multi-criteria decision making, the preference relation that arises from the pairwise comparisons may contain cycles. Such preferences are known as tournaments. Also, when the number of alternatives is large (e.g., in many human computation systems), it is hard for voters to submit a total order over the alternatives. Hence, most systems employ vote elicitation techniques where the individuals iteratively submit parts of their preference, such as pairwise comparisons or partial orders. Bounded rationality of voters may again lead to cyclic preferences in this case. There are also settings where a voter may in fact be a meta-voter, representing a group of individuals (e.g., a country or region). Synthesizing preferences of the people in a group may also lead to cyclic preference for the meta-voter.

Motivated by this, we introduce a variant of Mallows' model where both the ground truth and the noisy preferences generated may be tournaments rather than rankings of the alternatives. In this model, the vote generation process described above simplifies: there is no need to restart the process if the generated vote has cycles. Consequently, the pairwise comparisons are independent of each other, resulting in a more tractable model: indeed, it appears that Young [29, p. 1238] in his analysis of Condorcet's approach to choosing the most likely winner uses the tournament-based model in his calculations, even though his intention was to study the ranking-based model.

Recently, Procaccia et al. [22] have formalized, corrected, and extended Young’s analysis of the optimal rule to pick the best alternative. They focused on the limiting case of Mallows’ model where the noise is very high ($p \rightarrow 1/2$), as they were motivated by crowdsourcing settings, in which this is often the case. As a side result, they have also analyzed the other extreme case of very low noise ($p \rightarrow 1$).

Our Contribution: We introduce the tournament variant of Mallows’ model and show that the most likely winners in our model can be identified in polynomial time for a given value of the noise parameter, as well as in the limiting cases of extremely high and low noise; this is in sharp contrast with the ranking-based model. We then focus on the limiting cases of the tournament-based model and show empirically that they provide a good approximation for the corresponding cases of the ranking-based model. As a side result, we prove that Tideman’s rule [24, pp. 199–201] (which is closely related to the high-noise setting in the tournament-based model) is a 2-approximation of Kemeny’s rule (which is closely related to the high-noise setting in the ranking-based model), a result that may be of independent interest to the social choice community. Finally, we propose an agnostic voting rule that circumvents the problem of not knowing the noise parameter and returns the set of alternatives that are MLE at some value of the parameter. Using simulations, we show that this rule is quite decisive, i.e., returns very few alternatives.

2 RELATED WORK

The maximum likelihood estimation (MLE) approach to voting was proposed by Condorcet [11]. Young [29] formalized Condorcet’s ideas, and showed that Condorcet’s approach to choosing the best ranking results in a voting rule that is known as Kemeny’s rule. Young has also considered the problem of selecting the most likely winner, focusing on the limit cases where the noise is extremely high or extremely low. However, his analysis of this setting is presented by means of an example and appears to be flawed. Procaccia et al. [22] formalized Young’s analysis and extended it to objectives other than picking the best alternative. Recently, Caragiannis et al. [5] have further generalized this approach by focusing on the design of voting rules that demonstrate robustness to noise originating from a wide family of noise models. Such robustness is also a feature of our agnostic rule, in that it returns a set of alternatives that is guaranteed to contain the most likely alternative irrespective of the value of the underlying parameter of Mallows’ model from which votes are generated.

Other variants of the maximum likelihood approach have been considered in the computational social choice literature [9, 8, 27]. Perhaps the closest to our work is that of Xia et al. [28], who studied the MLE approach in multi-issue domains, where alternatives represent combinations

of multiple issues. Xia et al. used CP-nets to represent the (possibly cyclic) preferences of the voters. However, they focused on dealing with the huge space of alternatives created by an exponential number of combinations.

Finally, we show that Tideman’s rule provides a very simple and elegant deterministic 2-approximation to Kemeny’s rule. Since Kemeny’s rule is \mathcal{NP} -hard to compute [2], its approximations have been studied extensively in the literature, varying from deterministic approximations [10, 26] through randomized approximations [1] to a polynomial time approximation scheme (PTAS) [13].

3 PRELIMINARIES

Let $[k] = \{1, \dots, k\}$. We consider a set of *alternatives* A with $|A| = m$. Let $\mathcal{L}(A)$ denote the set of *votes*, where a vote is a ranking (linear order) over the alternatives, denoted $\sigma : \{1, \dots, m\} \rightarrow A$. Thus, alternative $\sigma(i)$ is the i -th most preferred alternative in σ ; $\sigma(1)$ and $\sigma(m)$ are, respectively, the most and the least preferred alternatives in σ . Note that $|\mathcal{L}(A)| = m!$. A *profile* $\pi \in \mathcal{L}(A)^n$ is a collection of n votes. For alternatives $a, b \in A$, let n_{ab} denote the number of votes in π that rank a above b . Hence, $n_{ab} + n_{ba} = n$ for all $a, b \in A$. Let $\Delta_{ab} = n_{ab} - n_{ba}$; this quantity can be thought of as the advantage of a over b .

Voting rules. A *voting rule* is a function that maps every profile to a *winning alternative*, or a set of tied winning alternatives. Formally, a voting rule is a mapping $f : \mathcal{L}(A)^n \rightarrow \mathcal{P}(A)$, where $\mathcal{P}(\cdot)$ denotes the power set.¹ We review three prominent voting rules that play a crucial role in this paper.

- *The Borda count.* Under the Borda count, each voter awards $m - i$ points to the alternative she ranks in position i , i.e., each alternative receives a number of points equal to the number of alternatives it defeats. The scores of the alternatives are tallied across the votes. That is, the Borda score of an alternative $a \in A$ in profile π is

$$\text{SC}^{BD}(a) = \sum_{\sigma \in \pi} \sum_{b \in A \setminus \{a\}} \mathbb{I}[a \succ_{\sigma} b] = \sum_{b \in A \setminus \{a\}} n_{ab},$$

where the second equality follows by switching the order of summation. The winner(s) are the alternative(s) with the highest score.

- *Tideman’s rule.* More commonly known as *Tideman’s simplified Dodgson rule*,² this rule was put forward by Tideman [24, pp. 199–201] as a polynomial-time computable approximation to Dodgson’s rule [12], which is

¹Technically, such mappings are known as *social choice functions*. In contrast, *social welfare functions* map every profile to a ranking or a set of tied rankings over the alternatives.

²Tideman’s rule considered in this paper should not be confused with *the ranked pairs method*, also proposed by Tideman.

\mathcal{NP} -hard to compute. Under Tideman’s rule, the score of an alternative a is given by

$$\text{SC}^{TD}(a) = \sum_{b \in A \setminus \{a\}} \max(0, \Delta_{ba}).$$

That is, the Tideman score of a is the cumulative advantage of all alternatives that have a positive advantage over a . Importantly, the winners are the alternatives with the *minimum* score.

- **Kemeny’s rule.** The Kendall tau distance between two rankings is the number of pairs of alternatives on which they disagree, i.e., for $\sigma_1, \sigma_2 \in \mathcal{L}(A)$, $d(\sigma_1, \sigma_2) = |\{(a, b) \mid a \succ_{\sigma_1} b, b \succ_{\sigma_2} a\}|$. With slight abuse of notation, for a profile π and a ranking σ , let $d(\pi, \sigma) = \sum_{\sigma' \in \pi} d(\sigma, \sigma')$. Under Kemeny’s rule, the score of an alternative $a \in A$ is the minimum distance from the input profile to any ranking that puts a first. Formally,

$$\text{SC}^{KM}(a) = \min_{\sigma \in \mathcal{L}(A): \sigma(1)=a} d(\pi, \sigma).$$

The winners are the alternatives with the *minimum* score. Equivalently, the rankings with the smallest distance from the profile are selected, and the winners are the alternatives appearing first in these rankings.

Refinement of voting rules. We say that voting rule \hat{f} is a *refinement* of voting rule f if $\hat{f}(\pi) \subseteq f(\pi)$ for every profile π . That is, \hat{f} can be seen as a combination of f with a (partial) tie-breaking rule.

4 MODEL

We begin by presenting the well-known ranking version of Mallows’ model, and then we introduce a more general tournament version of this model.

4.1 THE RANKING MODEL

Assume there is a hidden true ranking $\sigma^* \in \mathcal{L}(A)$ over the alternatives, which reflects the order of their true strengths. We also make the standard assumption that σ^* is selected using a uniform prior over $\mathcal{L}(A)$. Thus, $\sigma^*(1)$ denotes the true best alternative. A *noise model* describes how votes are generated given the true ranking. Votes in a profile are then assumed to be iid samples from the noise model.

Specifically, in Mallows’ model [18] (also known as the *Condorcet noise model* [11]), which was described informally in the introduction, the probability of generating a vote σ when the true ranking is σ^* is given by

$$\Pr[\sigma \mid \sigma^*] = \frac{\varphi^{d(\sigma, \sigma^*)}}{Z_\varphi^m}. \quad (1)$$

Here, $\varphi = \frac{1-p}{p} \in (0, 1)$ is the noise parameter of the model and $p \in (1/2, 1)$ is the probability of making the correct

decision when comparing two alternatives. Thus, $\varphi \rightarrow 0$ represents a distribution concentrated around σ^* , whereas $\varphi \rightarrow 1$ converges to the uniform distribution, which has the greatest noise. Finally, d is the Kendall tau distance, and $Z_\varphi^m = \sum_{\sigma \in \mathcal{L}(A)} \Pr[\sigma \mid \sigma^*]$ is the normalization constant, which turns out to be independent of the true ranking σ^* (see, e.g., [16]).

Now, take a profile $\pi \in \mathcal{L}(A)^n$. Since individual votes are sampled iid, the probability of generating π is

$$\Pr[\pi \mid \sigma^*] = \prod_{\sigma \in \pi} \frac{\varphi^{d(\sigma, \sigma^*)}}{Z_\varphi^m} \propto \varphi^{d(\pi, \sigma^*)}.$$

Under the assumption of uniform prior over the true ranking σ^* , and for given φ , the probability of an alternative $a \in A$ being the true best alternative is proportional to

$$\sum_{\substack{\sigma^* \in \mathcal{L}(A): \\ \sigma^*(1)=a}} \Pr[\pi \mid \sigma^*] \propto \sum_{\substack{\sigma^* \in \mathcal{L}(A): \\ \sigma^*(1)=a}} \varphi^{d(\pi, \sigma^*)}. \quad (2)$$

Let $\Gamma_\varphi^R(a)$ be the “likelihood polynomial” of a , as given in the final expression of Equation (2). Then, the *maximum likelihood estimator* of the true best alternative, i.e., the set of alternatives having the highest probability of being the true best alternative, is given by $\text{MLE}_\varphi^R(\pi) = \arg \max_{a \in A} \Gamma_\varphi^R(a)$. Theorem 3.2 by Procaccia et al. [22] shows the following.

Theorem 1 (Procaccia et al. [22]). *Computing MLE_φ^R is \mathcal{NP} -hard.*

4.2 THE TOURNAMENT MODEL

We now introduce a variant of Mallows’ model where both the ground truth and the samples need not be total orders. Rather, they can be *tournaments*, i.e., sets of pairwise comparisons (one for each pair of alternatives). A tournament need not be transitive: it can be the case that a beats b , b beats c , and c beats a . As argued in the introduction, this is common when the alternatives are compared based on multiple quality parameters instead of a single parameter, and/or users are not required to submit total orders. Note that every ranking can be seen as a tournament.

Let $\mathcal{T}(A)$ denote the set of all tournaments over alternatives in A . We still use $a \succ_T b$ to denote that alternative a is preferred to alternative b in the tournament T . The Kendall tau distance extends to $\mathcal{T}(A)$ in a natural way: given two tournaments $T, T' \in \mathcal{T}(A)$, $d(T, T')$ is the number of pairs of alternatives on which T and T' disagree. Further, the quantities $(n_{ab})_{a,b \in A}$ remain well-defined for a profile of tournaments $\pi \in \mathcal{T}(A)^n$. As the three voting rules introduced in Section 3 (the Borda count, Tideman’s rule, and Kemeny’s rule) can be defined in terms of $(n_{ab})_{a,b \in A}$, these rules are well-defined over profiles of tournaments as well.

Let $T^* \in \mathcal{T}(A)$ denote the hidden true tournament over the alternatives. We assume that T^* is selected using a uniform prior over $\mathcal{T}(A)$. That is, for each pair of alternatives $a, b \in A$ we independently decide whether $a \succ_{T^*} b$ or $b \succ_{T^*} a$, with both possibilities being equally likely. When generating a vote, each pairwise comparison in T^* is retained with a fixed probability $1/2 < p < 1$ and flipped with probability $1 - p$. Unlike in the ranking model, the pairwise comparisons in the samples are independent of each other. Accordingly, the probability of generating a tournament T when the true tournament is T^* is given by

$$\Pr[T \mid T^*] = p^{\binom{m}{2} - d(T, T^*)} (1-p)^{d(T, T^*)} = p^{\binom{m}{2}} \varphi^{d(T, T^*)}.$$

We consider profiles consisting on n tournaments, which are sampled iid from the noise model. Let $d(\pi, T^*) = \sum_{T \in \pi} d(T, T^*)$. Then, the probability of generating a profile $\pi \in \mathcal{T}(A)^n$ is proportional to $\varphi^{d(\pi, T^*)}$, similarly to the ranking-based model.

Procaccia et al. [22] introduced the noisy choice model as the generalization of Mallows' model where the ground truth was a ranking but the samples could be tournaments. In that sense, our model is a further generalization where even the ground truth may be a tournament.

However, this causes a potentially serious problem: The best alternative in a ranking σ^* is $\sigma^*(1)$. But the definition of the best alternative in a tournament T^* is unclear. Following Condorcet's own definition of "Condorcet winners" for cyclic majority preferences, we say that an alternative is the winner in a tournament if it is preferred to every other alternative. Note that not every tournament has a winner. For a tournament T , define $\text{win}(T)$ to be the winner of T if it exists, and \emptyset otherwise.

Given a profile $\pi \in \mathcal{T}(A)^n$, we can now compute the likelihood of an alternative $a \in A$ being the best alternative in the unknown true tournament. Indeed, for every $T^* \in \mathcal{T}(A)$ with $\text{win}(T^*) = a$ we have

$$\begin{aligned} d(\pi, T^*) &= \sum_{b \in A \setminus \{a\}} n_{ba} \\ &+ \sum_{c, d \in A \setminus \{a\}} (n_{cd} \cdot \mathbb{I}[d \succ_{T^*} c] + n_{dc} \cdot \mathbb{I}[c \succ_{T^*} d]). \end{aligned}$$

Further, for each possible combination of pairwise comparisons of the alternatives in $A \setminus \{a\}$, the set $\{T^* \in \mathcal{T}(A) \mid \text{win}(T^*) = a\}$ contains exactly one tournament that realizes this combination. Hence, we have

$$\begin{aligned} \sum_{\substack{T^* \in \mathcal{T}(A): \\ \text{win}(T^*) = a}} \varphi^{d(\pi, T^*)} &= \varphi^{\sum_{b \in A \setminus \{a\}} n_{ba}} \cdot \prod_{c, d \in A \setminus \{a\}} (\varphi^{n_{cd}} + \varphi^{n_{dc}}) \\ &\propto \prod_{b \in A \setminus \{a\}} \frac{\varphi^{n_{ba}}}{\varphi^{n_{ba}} + \varphi^{n_{ab}}} = \prod_{b \in A \setminus \{a\}} \frac{1}{1 + \varphi^{n_{ab} - n_{ba}}}. \end{aligned}$$

Now, for an alternative $a \in A$, define its likelihood polynomial $\Gamma_\varphi^T(a) = \prod_{b \in A \setminus \{a\}} (1 + \varphi^{n_{ab} - n_{ba}})$. Technically,

$\Gamma_\varphi^T(a)$ is a Laurent polynomial, i.e., some of the powers of φ may be negative. Therefore, we will sometimes work with the function $\hat{\Gamma}_\varphi^T(a) = \varphi^{nm} \Gamma_\varphi^T(a)$, which is a polynomial of degree at most $2nm$. Note that the likelihood polynomial of a is proportional to the inverse of the likelihood, and is therefore to be minimized. Thus, the maximum likelihood estimator for the best alternative is given by $\text{MLE}_\varphi^T(\pi) = \arg \min_{a \in A} \Gamma_\varphi^T(a)$, or, equivalently, $\text{MLE}_\varphi^T(\pi) = \arg \min_{a \in A} \hat{\Gamma}_\varphi^T(a)$. Since $\Gamma_\varphi^T(a)$ can be computed for every alternative $a \in A$ and every $\varphi \in \mathbb{Q}$ in polynomial time, the following is trivial.

Theorem 2. *Computing MLE_φ^T is in \mathcal{P} .*

5 LIMITING VOTING RULES

We will now study the extreme cases with very low and very high noise in input votes, i.e., $\varphi \rightarrow 0$ and $\varphi \rightarrow 1$, respectively. First, we observe that both for rankings and for tournaments and in both limiting cases, the limiting rule is well-defined, i.e., there exist α and β with $0 < \alpha < \beta < 1$ such that for $P \in \{R, T\}$ $\text{MLE}_\varphi^P = \text{MLE}_\alpha^P$ for all $0 < \varphi \leq \alpha$ and $\text{MLE}_\varphi^P = \text{MLE}_\beta^P$ for all $\beta \leq \varphi < 1$. Indeed, fix a profile π . For each $a \in A$ the degree of the likelihood polynomials $\Gamma_\varphi^R(a)$ and $\hat{\Gamma}_\varphi^T(a)$ is finite, and therefore we can pick $\alpha^\pi, \beta^\pi \in (0, 1)$ so that no two of these polynomials for π intersect in $(0, \alpha^\pi)$ or in $(\beta^\pi, 1)$. Since the number of profiles with a *fixed number of votes* is finite, taking the minimum of α^π and the maximum of β^π over all such profiles gives the desired values of α and β . Note, however, that this argument breaks down if the number of votes may vary. For example, if $\Gamma_\varphi^R(a)$ and $\Gamma_\varphi^R(b)$ for a profile π intersect at φ , then $\Gamma_\varphi^R(a)$ and $\Gamma_\varphi^R(b)$ for the profile $k\pi$ intersect at $\sqrt[k]{\varphi}$ (where $k\pi$ is the profile where each entry of π is repeated k times). As k is unbounded, we obtain $\beta = \sup_\pi \beta^\pi = 1$.

Notation. For the ranking model, let $\text{MLE}_{\text{Acc}}^R$ and $\text{MLE}_{\text{Inacc}}^R$ denote the limiting rules in the accurate case ($\varphi \rightarrow 0$) and in the inaccurate case ($\varphi \rightarrow 1$), respectively. Similarly, for the tournament model, let $\text{MLE}_{\text{Acc}}^T$ and $\text{MLE}_{\text{Inacc}}^T$ denote the limiting rules in the accurate case and in the inaccurate case, respectively.

5.1 THE RANKING MODEL

The accurate case ($\varphi \rightarrow 0$): Procaccia et al. [22] showed that when $\varphi \rightarrow 0$, every MLE best alternative is first in some Kemeny ranking. Further, they also showed that finding even a single Kemeny winner is \mathcal{NP} -hard.³ These results can be restated as follows.

Theorem 3 (Procaccia et al. [22]). *$\text{MLE}_{\text{Acc}}^R$ is a refinement of Kemeny's rule, and is \mathcal{NP} -hard to compute.*

³Both results can be found in the proof of Theorem A.1 in the appendix of the full version available at <http://www.cs.cmu.edu/~arielp/papers/mle.full.pdf>.

In fact, our tools enable us to describe $\text{MLE}_{\text{Acc}}^R$ in more detail; see Appendix A of the full version of the paper.⁴

The inaccurate case ($\varphi \rightarrow 1$): Procaccia et al. [22, Theorem 4.1] proved that every MLE best alternative in this case is also a Borda winner. However, they left open the question of computational complexity. Despite significant effort, we were unable to settle the computational complexity either. We conjecture that $\text{MLE}_{\text{Inacc}}^R$ is \mathcal{NP} -hard to compute.

Theorem 4 (Procaccia et al. [22]). $\text{MLE}_{\text{Inacc}}^R$ is a refinement of the Borda count.

Once again, the exact refinement is given in Appendix A of the full version. While the computational complexity of $\text{MLE}_{\text{Inacc}}^R$ is unknown, we remark that computing its output is easy whenever Borda’s rule produces a unique winner, which is often the case.

See Section 5.3 for an example showing the computation of $\text{MLE}_{\text{Acc}}^R$ and $\text{MLE}_{\text{Inacc}}^R$ for a given profile.

5.2 THE TOURNAMENT MODEL

The accurate case ($\varphi \rightarrow 0$): In this case we show the following.

Theorem 5. $\text{MLE}_{\text{Acc}}^T$ is a refinement of Tideman’s rule, and can be computed in polynomial time.

Proof. To determine the winner(s) under $\text{MLE}_{\text{Acc}}^T$ we need to compare the likelihood polynomials $\Gamma_\varphi^T(a) = \prod_{b \in A \setminus \{a\}} (1 + \varphi^{n_{ab} - n_{ba}})$ when $\varphi \rightarrow 0$. Pick $\alpha \in (0, 1)$ so that no two likelihood polynomials intersect in $(0, \alpha)$. As $\varphi \rightarrow 0$, the dominating term in $\Gamma_\varphi^T(a)$ is the smallest power of φ , i.e.,

$$t_\varphi(a) = \prod_{\substack{b \in A \setminus \{a\}, \\ n_{ab} \leq n_{ba}}} \varphi^{n_{ab} - n_{ba}} = \varphi^{-\sum_{b \in A \setminus \{a\}} \max\{0, \Delta_{ba}\}}$$

(where we take the product over the empty set to be 1). Hence, for $\varphi \in (0, \alpha)$ we have $\Gamma_\varphi^T(a) < \Gamma_\varphi^T(b)$ whenever $t_\varphi(a) < t_\varphi(b)$, or, equivalently, whenever $\text{SC}^{TD}(a) > \text{SC}^{TD}(b)$. Recall that we are interested in alternatives with the smallest value of the likelihood polynomial on $(0, \alpha)$; our calculation shows that every such alternative is a Tideman winner.

To show that $\text{MLE}_{\text{Acc}}^T$ is polynomial-time computable, it is not sufficient to observe that the functions $\Gamma_\varphi^T(a)$, $a \in A$, can be evaluated in polynomial time, as we also need to find a small enough value of φ at which they should be compared. Nevertheless, comparing likelihood polynomials at $\varphi \rightarrow 0$ is not difficult. We first multiply the terms of $\hat{\Gamma}_\varphi^T(a)$ one-by-one, followed by expansion at each stage,

to obtain the coefficients of this polynomial. Note that the degree of $\hat{\Gamma}_\varphi^T(a)$ is at most $2mn$, so this step can be implemented efficiently. To compare two polynomials at $\varphi \rightarrow 0$, it suffices to consider their coefficients lexicographically, starting with the lowest-order terms. The details are given in Appendix A of the full version of the paper. \square

The inaccurate case ($\varphi \rightarrow 1$): This case has striking similarity with the inaccurate case of the ranking model.

Theorem 6. $\text{MLE}_{\text{Inacc}}^T$ is a refinement of the Borda count, and can be computed in polynomial time.

Proof. Note that $\hat{\Gamma}_1^T(a) = \Gamma_1^T(a) = 1$ for all $a \in A$. Therefore, to compare the likelihood polynomials as $\varphi \rightarrow 1$, we will first compare their derivatives at $\varphi = 1$. We have

$$\begin{aligned} \left. \frac{d}{d\varphi} \Gamma_\varphi^T(a) \right|_{\varphi=1} &= 2^{m-2} \sum_{b \in A \setminus \{a\}} \left. \frac{d}{d\varphi} (1 + \varphi^{n_{ab} - n_{ba}}) \right|_{\varphi=1} \\ &= 2^{m-2} \sum_{b \in A \setminus \{a\}} (n_{ab} - n_{ba}). \end{aligned}$$

As φ approaches 1 from the left, we have $\Gamma_\varphi^T(a) < \Gamma_\varphi^T(b)$ whenever $\left. \frac{d}{d\varphi} \Gamma_\varphi^T(a) \right|_{\varphi=1} > \left. \frac{d}{d\varphi} \Gamma_\varphi^T(b) \right|_{\varphi=1}$. Using $n_{ba} = n - n_{ab}$, we observe that the latter condition is equivalent to $\text{SC}^{BD}(a) > \text{SC}^{BD}(b)$. Thus, the winners under $\text{MLE}_{\text{Inacc}}^T$ must have the highest Borda score, i.e., $\text{MLE}_{\text{Inacc}}^T$ is a refinement of the Borda rule.

In contrast with the ranking-based model, the rule $\text{MLE}_{\text{Inacc}}^T$ can be computed in polynomial time. Similarly to the accurate case of the tournament model (see the proof of Theorem 5), we multiply the terms of each $\hat{\Gamma}_\varphi^T(a)$, $a \in A$, in order to obtain the coefficients of these polynomials. Then, for each polynomial we compute its first $2nm$ derivatives at $\varphi = 1$. As the degree of each of these polynomials does not exceed $2nm$, comparing two such polynomials at $\varphi \rightarrow 1$ amounts to lexicographically comparing these two lists of values. \square

Alternatively, we can show that $\text{MLE}_{\text{Acc}}^T$ and $\text{MLE}_{\text{Inacc}}^T$ are polynomial-time computable by using results on *root separation* of polynomials. A classic paper by Mahler [17] proved the following.

Fact: Any two distinct roots of a polynomial are separated by at least H^{-k+1} , where H is the maximum absolute value of any coefficient, and k is the degree.

See [3] for further explanation and improved results for polynomials with integer coefficients. In our case, we can consider the polynomial $P = \prod_{a, b \in A} (\hat{\Gamma}_\varphi^T(a) - \hat{\Gamma}_\varphi^T(b))$; its degree does not exceed $2nm^3$ and its coefficients are at most exponential in $\text{poly}(n, m)$. Thus, $\Delta = H^{-k+1}$ has *polynomially many bits*. An exponential upper bound on H (and hence the respective lower bound on Δ) can be

⁴The full version can be found at <http://www.cs.cmu.edu/~nkshah/papers.html>

computed without expanding P . Since no two likelihood polynomials intersect on $(0, \Delta)$ or on $(1 - \Delta, 1)$, the outputs of $\text{MLE}_{\text{Acc}}^T$ and $\text{MLE}_{\text{Inacc}}^T$ can be computed by evaluating and comparing $\Gamma_{\varphi}^T(a)$, $a \in A$, in their product form at $\varphi = \Delta/2$ and $\varphi = 1 - \Delta/2$, respectively. We prefer the approach presented in the proofs above because it seems to work faster in practice, possibly due to the fact that it relies only on integer arithmetic.

5.3 THE TOURNAMENT MODEL VERSUS THE RANKING MODEL

The goal of this section is to compare the ranking-based model and the tournament-based model. We begin by presenting a profile on which the limiting voting rules for the two models differ.

Example 1. Consider a profile π consisting of the following 3 rankings over 4 alternatives.

$$a \succ b \succ c \succ d, \quad d \succ a \succ b \succ c, \quad c \succ d \succ b \succ a.$$

Recall that the likelihood polynomials $\Gamma_{\varphi}^R(a)$ (in the ranking model) and $\Gamma_{\varphi}^T(a)$ (in the tournament model) of an alternative $a \in A$ are given by

$$\begin{aligned} \Gamma_{\varphi}^R(a) &= \sum_{\substack{\sigma^* \in \mathcal{L}(A): \\ \sigma^*(1)=a}} \varphi^{d(\pi, \sigma^*)}, \\ \Gamma_{\varphi}^T(a) &= \prod_{b \in A \setminus \{a\}} (1 + \varphi^{n_{ab} - n_{ba}}). \end{aligned}$$

For profile π , the likelihood polynomials are given below.

$$\begin{aligned} \Gamma_{\varphi}^R(a) &= \varphi^8 + \varphi^8 + \varphi^8 + \varphi^9 + \varphi^9 + \varphi^9, \\ \Gamma_{\varphi}^R(b) &= \varphi^9 + \varphi^9 + \varphi^9 + \varphi^{10} + \varphi^{10} + \varphi^{10}, \\ \Gamma_{\varphi}^R(c) &= \varphi^8 + \varphi^9 + \varphi^9 + \varphi^{10} + \varphi^{10} + \varphi^{11}, \\ \Gamma_{\varphi}^R(d) &= \varphi^7 + \varphi^8 + \varphi^8 + \varphi^9 + \varphi^9 + \varphi^{10}, \\ \Gamma_{\varphi}^T(a) &= (1 + \varphi^1) (1 + \varphi^1) (1 + \varphi^{-1}), \\ \Gamma_{\varphi}^T(b) &= (1 + \varphi^{-1}) (1 + \varphi^1) (1 + \varphi^{-1}), \\ \Gamma_{\varphi}^T(c) &= (1 + \varphi^{-1}) (1 + \varphi^{-1}) (1 + \varphi^1), \\ \Gamma_{\varphi}^T(d) &= (1 + \varphi^1) (1 + \varphi^1) (1 + \varphi^{-1}). \end{aligned}$$

Now, we can compute the limiting voting rules using the likelihood polynomials as explained in Sections 5.1 and 5.2. The results of these rules along with those of the Borda count, Kemeny's rule, and Tideman's rule are given in Table 1.

While Example 1 shows that the accurate and the inaccurate cases of the tournament model differ from the respective cases of the ranking model, we show that the tournament model serves as a satisfactory polynomial-time approximation of the ranking model, where computing the limiting rules is non-trivial (and provably \mathcal{NP} -hard in the

Ranking, Accurate	$\text{MLE}_{\text{Acc}}^R(\pi) = \{d\}$
Ranking, Inaccurate	$\text{MLE}_{\text{Inacc}}^R(\pi) = \{d\}$
Tournament, Accurate	$\text{MLE}_{\text{Acc}}^T(\pi) = \{a, d\}$
Tournament, Inaccurate	$\text{MLE}_{\text{Inacc}}^T(\pi) = \{a, d\}$
Borda count	$\text{Borda}(\pi) = \{a, d\}$
Kemeny's rule	$\text{Kemeny}(\pi) = \{d\}$
Tideman's rule	$\text{Tideman}(\pi) = \{a, d\}$

Table 1: Various voting rules applied on π .

accurate case). While the tournament model—where both the ground truth and the estimates may be cyclic—has its intrinsic motivation (see Section 1), this offers an additional strong motivation for the model. The similarity of both models in the inaccurate case is evident: Both $\text{MLE}_{\text{Inacc}}^R$ and $\text{MLE}_{\text{Inacc}}^T$ are refinements of Borda's rule. However, as seen in Example 1, these two rules are not identical. Moreover, we can show that neither of these rules is a refinement of the other: Appendix C in the full version presents an example with 8 rankings over 4 alternatives where both $\text{MLE}_{\text{Inacc}}^R$ and $\text{MLE}_{\text{Inacc}}^T$ have unique winners that are different. We remark, however, that these two rules return the same output most of the time (see Section 7), and always when the Borda winner is unique.

Motivated by the similarity in the inaccurate case, we compared Tideman's rule and Kemeny's rule, because the limiting rules in the accurate case of the ranking and the tournament models are refinements of Kemeny's rule and Tideman's rule, respectively. While the two rules were not previously thought to be connected, we show that Tideman's rule is a 2-approximation of Kemeny's rule.

Theorem 7. *The Kemeny score of a Tideman winner is at most twice the Kemeny score of a Kemeny winner.*

Proof. First, we describe an alternative interpretation of Kemeny's rule proposed by Conitzer et al. [7]. Given a profile π over A , define a *weighted pairwise majority (WPM) graph* for a set of alternatives $A' \subseteq A$ to be the directed graph $G_{A'}$ where the vertices are the alternatives in A' , and there is an edge between every pair of alternatives $a, b \in A'$ with weight $|\Delta_{ab}| = |n_{ab} - n_{ba}|$. The edge goes from a to b if $n_{ab} > n_{ba}$ and from b to a if $n_{ba} > n_{ab}$. When $n_{ab} = n_{ba}$, the edge with zero weight may be drawn in either direction.

The *feedback* of a ranking with respect to a WPM graph $G_{A'}$ is defined as the sum of the weights of edges of $G_{A'}$ going in direction opposite to the ranking. Conitzer et al. [7] showed that Kemeny's rule is equivalent to first finding the rankings with the smallest feedback with respect to G_A , and then returning their top alternatives.

Equivalently, we can say that in G_A for every pair of alternatives $a, b \in A$ there is an edge from a to b with weight $\max(0, \Delta_{ab})$. Thus, the Tideman score of an alternative

$a \in A$ is the sum of weights of its incoming edges, and the Tideman winners are the vertices that minimize this sum. For a subset of alternatives $S \subseteq A$, let $F(S)$ be the smallest feedback with respect to G_S , over all rankings of A . To compute the Kemeny score of an alternative $a \in A$, we consider the set of all rankings \mathcal{L}_a that put a first, and find a ranking in \mathcal{L}_a that has the minimum feedback with respect to G_A . Note that the feedback of any ranking in \mathcal{L}_a contains all incoming edges of a . Thus, to minimize feedback over \mathcal{L}_a , we order the alternatives in $A \setminus \{a\}$ so as to minimize the feedback over $G_{A \setminus \{a\}}$. Hence,

$$\text{SC}^{KM}(a) = \text{SC}^{TD}(a) + F(A \setminus \{a\}). \quad (3)$$

Further, for every pair of alternatives $a, b \in A$, $a \neq b$, we have

$$F(A \setminus \{a\}) \leq \text{SC}^{TD}(b) + F(A \setminus \{a, b\}). \quad (4)$$

Indeed, the left-hand side of (4) is the feedback of the best ranking with respect to $G_{A \setminus \{a\}}$, whereas the right-hand side of (4) is the feedback of the best ranking with respect to $G_{A \setminus \{a\}}$ among those that put b first, plus $\max(0, \Delta_{ab})$.

Now, consider a profile π . Let $a \in A$ be a Tideman winner and let $b \in A$ be a Kemeny winner. We want to show that $\text{SC}^{KM}(a) \leq 2\text{SC}^{KM}(b)$. If $a = b$, this is trivial. Thus, assume $a \neq b$. Combining (3) and (4), we obtain

$$\begin{aligned} \text{SC}^{KM}(a) &= \text{SC}^{TD}(a) + F(A \setminus \{a\}) \\ &\leq \text{SC}^{TD}(a) + \text{SC}^{TD}(b) + F(A \setminus \{a, b\}) \\ &\leq \text{SC}^{TD}(b) + \text{SC}^{TD}(b) + F(A \setminus \{b\}) \\ &\leq 2(\text{SC}^{TD}(b) + F(A \setminus \{b\})) = 2 \cdot \text{SC}^{KM}(b). \end{aligned}$$

Hence, the Kemeny score of any Tideman winner is a 2-approximation of the optimal Kemeny score. \square

It is easy to give an example using 4 alternatives where the approximation factor is exactly 2. Hence, the result of Theorem 7 is tight. Caragiannis et al. [4] show that Tideman's rule, which was originally proposed as an approximation to Dodgson's rule, is actually an asymptotically optimal approximation of Dodgson's rule. Theorem 7 shows that it is also a 2-approximation of Kemeny's rule. Dodgson's rule and Kemeny's rule are deeply connected [23]: While Dodgson's rule makes the smallest number of pairwise swaps to reach a profile with a Condorcet winner (an alternative preferred by a majority of voters to every other alternative), Kemeny's rule makes the swaps until the majority opinion becomes acyclic and then returns the first alternative in the acyclic order. Tideman's rule can now be seen as a hybrid that provides a good approximation to both Dodgson's rule and Kemeny's rule.

Kemeny's rule admits a polynomial time approximation scheme (PTAS) [13], which is, however, rather impractical. Constant approximations of Kemeny's rule are studied

because they are fast and simple [10, 1, 26]. Tideman's rule, which admits an elegant closed-form expression (see Section 3), is the simplest deterministic 2-approximation of Kemeny's rule that we are aware of. We believe that this result, which originated from the observation that the tournament model seems closely related to the ranking model, may be of independent interest to the social choice community.

6 THE AGNOSTIC RULE

While the limiting cases of $\varphi \rightarrow 0$ and $\varphi \rightarrow 1$ may be appropriate in some scenarios (and their analysis yields interesting connections, e.g., Theorem 7), in most practical settings the level of noise is unknown. We could include φ as one of the unknown parameters and infer the best possible values for the true ranking/tournament and φ (see, e.g., [16]). However, this approach gives one specific value (a "point estimate") of φ . If the point estimate is wrong, the estimate for the best alternative is also sub-optimal.

Consider instead an agnostic approach that refrains from estimating the value of φ . Rather, given a profile, it returns the set of *all* alternatives that are the most likely winners for *some value of* φ . Let MLE_{Ag}^R and MLE_{Ag}^T denote the agnostic rules in the ranking model and in the tournament model, respectively. Then for a profile π ,

$$\begin{aligned} \text{MLE}_{\text{Ag}}^R(\pi) &= \bigcup_{\varphi \in (0,1)} \text{MLE}_{\varphi}^R(\pi), \\ \text{MLE}_{\text{Ag}}^T(\pi) &= \bigcup_{\varphi \in (0,1)} \text{MLE}_{\varphi}^T(\pi). \end{aligned}$$

There are three advantages of this approach over the inference approach: First, as we show below, MLE_{Ag}^T can be computed in polynomial time, and results presented in Section 7 demonstrate that MLE_{Ag}^T is a good approximation of MLE_{Ag}^R as well. Thus, it is easy to compute and use the agnostic rule. Most inference problems, on the other hand, are hard to solve [16]. Second, if the data is indeed generated from a Mallows' (ranking or tournament) model, the MLE best alternative is guaranteed to be in the set returned by the agnostic rule, which is not the case for the inference approach. Third, while the set of winners under the agnostic rules contains the set of winners for any specific value of φ , our simulations in Section 7 show that on average only a few winning alternatives are returned.

Now, we show that the agnostic rule can be computed in polynomial time for the tournament model. Note that this is not obvious: While the limiting cases can be analyzed by looking at the coefficients of the likelihood polynomials or using a root separation approach, these methods do not work for values of φ away from 0 and 1. We use another result regarding polynomials, which is known as root isolation (see, e.g., [6, 25]).

Fact: Given a polynomial, one can compute, in time polynomial in the input size, a set of disjoint intervals that isolate the roots of the polynomial, i.e., disjoint intervals that collectively contain all roots of the polynomial but each interval only contains a single distinct root.

Theorem 8. *Computing MLE_{Ag}^T is in \mathcal{P} .*

Proof. Once again, consider the polynomial $P = \prod_{a,b \in A} (\hat{\Gamma}_{\varphi}^T(a) - \hat{\Gamma}_{\varphi}^T(b))$. We have argued that the degree of P is at most $2nm^3$, and its coefficients can be computed in polynomial time. Next, we use root isolation to isolate the roots of P in polynomial time. Note that any value of $\varphi \in (0, 1)$ where some alternatives a, b , $a \neq b$, have equal likelihood is a root of P . Hence, in any region between two consecutive roots of P , the order of likelihoods of different alternatives is fixed.

Therefore, computing MLE_{Ag}^T amounts to taking one value of φ between each consecutive pair of isolating intervals, evaluating MLE_{φ}^T at every such φ , and returning the set of all MLE alternatives found. Note that the number of roots of P and therefore the number of evaluations of MLE_{φ}^T is polynomial in the input size, and we have already established that evaluating MLE_{φ}^T itself can be done in polynomial time (Theorem 2). Hence, the overall running time is polynomial in the input size. \square

7 EXPERIMENTS

In this section, we complement our theoretical results by two sets of experiments. The results of Section 5.3 establish that the tournament model can be thought of as a polynomial-time approximation to the ranking model. The first set of experiments analyzes how close the limiting rules (and the agnostic rules) in the two models are to each other on average. The second set of experiments aims to check whether the agnostic rules return reasonably small sets of winning alternatives. To this end, we compute the average number of winning alternatives returned by the agnostic rules in the two models.

For both sets of experiments, we generate profiles using iid samples from Mallows' (ranking) model with the noise parameter φ taking 10 different values from 0.1 to 1.⁵ In each case, we average our results over 5000 sampled profiles. It is easy to check that the Borda winner, the Tideman winner, and the Kemeny winner always coincide in case of 3 alternatives. Hence, in our experiments we set the number of alternatives m to 5 or 7; the number of votes n is also either 5 or 7. Thus, each of the graphs presented has four lines; one for each of $(n, m) = (5, 5), (5, 7), (7, 5), (7, 7)$.

⁵We use the ranking model to generate profiles so that the generated profiles consist of rankings, which are also tournaments. This allows applying the rules for both models on the same profiles.

and (7, 7). In all the graphs, the x -axis shows the noise parameter φ used to generate profiles. Importantly, while the tournament model admits polynomial-time algorithms, and can therefore be used with a large number of alternatives, we use a small number of alternatives to be able to compare the associated voting rules with the exponential-time rules of the ranking model.

For the first set of experiments, we measure the dissimilarity between three pairs of rules in terms of the dissimilarity between the sets of winning alternatives they return. As the measure of dissimilarity between two sets A and B , we use the *Jaccard distance*, which is defined as follows.

$$d_J(A, B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|}.$$

Figures 1(a), 1(b), and 1(c) respectively show the dissimilarity between the limiting rules for the accurate case, the limiting rules for the inaccurate case, and the agnostic rules of the two models—the ranking model and the tournament model—as a function of their noise parameter φ (note that, while the limiting rules were derived for a specific range of the noise parameter φ , we compare them at *all* values of φ). An interesting observation is that while the dissimilarity is quite low in both the accurate and the agnostic cases, it is surprisingly low in the inaccurate case. This observation holds true for all combinations of (n, m) . This indicates that the MLE rules of the tournament model are in general good approximations of the MLE rules of the ranking model, and the approximation becomes very good for the rules derived under the assumption of very high noise.

Continuing our comparison of the two models, note that Theorem 7 establishes that Tideman's rule is a 2-approximation of Kemeny's rule in the worst case (in terms of the Kemeny score of the winner). This is a significant improvement over the Borda count, which is known to give a 4-approximation in the worst case [10]. Thus, Tideman's rule improves over Borda's rule by a factor of 2 in the worst case. It is interesting to check if this relationship holds even in the average case.

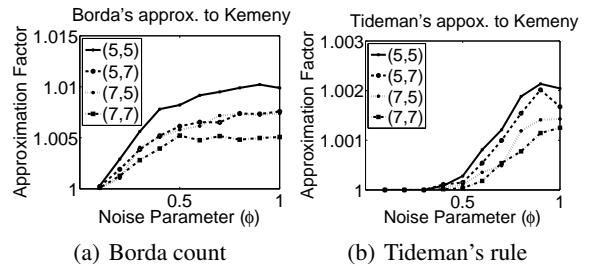


Figure 2: Approximations of Kemeny's rule.

Figures 2(a) and 2(b) show the average-case approximation factors of the Borda count and Tideman's rule, respectively, as a function of the noise parameter φ . It is evident

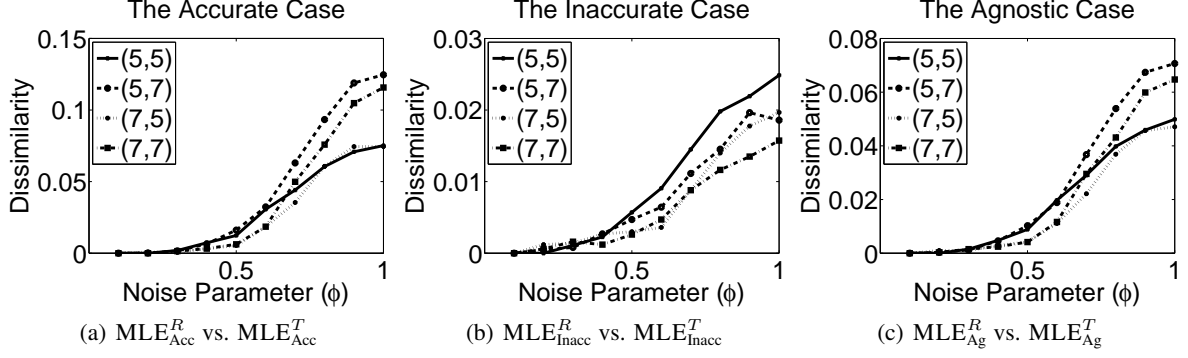


Figure 1: The dissimilarity between the ranking model and the tournament model.

that for both rules their average-case approximation factors are much better than their worst-case approximation factors. However, in the average case, the Borda count quickly reaches an approximation ratio of 1.01, while the approximation ratio of Tideman’s rule stays well below 1.003. That is, the improvement of Tideman’s rule over the Borda count is at least as good—in fact, slightly better—in the average case as in the worst case.

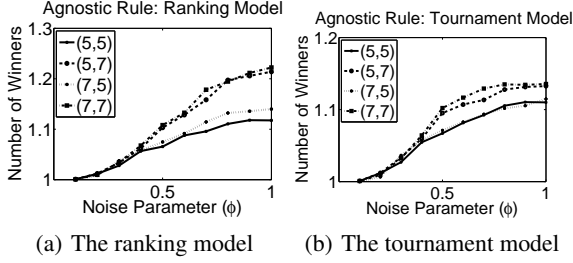


Figure 3: The average number of winning alternatives returned by the agnostic rules.

In our second set of experiments, we analyze the average number of winning alternatives returned by the agnostic rules in the ranking and the tournament models, again as a function of the noise parameter φ . Figures 3(a) and 3(b) show the results for the ranking and the tournament models, respectively. It can be seen that the agnostic rule—despite returning a set of alternatives that is guaranteed to contain the MLE best alternative for all values of $\varphi \in (0, 1)$ —outputs an average of less than 1.3 and 1.2 alternatives in the ranking model and the tournament model, respectively. In fact, in our simulations, the agnostic rules in both models return a single alternative, which is guaranteed to be the MLE best alternative for all values of φ , more than 80% of the time, for every $\varphi \in (0, 1)$.

8 DISCUSSION

We have studied methods for picking the best alternative given noisy estimates of an objective true comparison be-

tween the alternatives. Besides studying the standard Mallows’ model where both the ground truth and the estimates are acyclic total orders, we introduced and studied the setting where both may contain cycles. Procaccia et al. [22] studied the case where the ground truth is acyclic, but the estimates may or may not be cyclic. The only case not studied in the literature is that of possibly cyclic ground truth and acyclic estimates. However, this setting does not appear natural, and is also technically challenging: the denominator Z_φ^m in the probability expression of Mallows’ model (Equation 1) would not be independent of the ground truth, rendering the analysis extremely difficult.

Generalizations of Mallows’ model have been proposed in the literature [15, 20]. Some of these use critical information regarding positions of the alternatives in the ground truth ranking. Future work may also involve adapting such models to the case of tournaments; for example, one can use the number of alternatives defeated by a given alternative as a proxy for its rank, or one can develop distance metrics over tournaments to replace the Kendall tau distance in Equation (1). It would be interesting to see if such adaptations provide tractable approximations of the original ranking model.

The maximum likelihood approach to voting focuses solely on maximizing the likelihood of selecting the best alternative. This results in voting rules that can be difficult to understand, but have performance guarantees nonetheless. While simplicity is usually an important goal in the design of voting rules, it is less of an issue in human computation contexts, where the workers are paid for their input and do not need to know or understand how their estimates would be aggregated. Yet, in some practical applications, one may wish to use rules with additional desirable properties, either motivated by the application itself or as a safeguard in case the assumptions about the nature of the noise fail. A very exciting direction is to use Mallows’ model in order to inform the design of voting rules while trading off some of the likelihood for axiomatic properties such as Condorcet consistency or monotonicity.

References

- [1] N. Ailon, M. Charikar, and A. Newman. Aggregating inconsistent information: Ranking and clustering. In *Proc. of 37th STOC*, pages 684–693, 2005.
- [2] J. Bartholdi, C. A. Tovey, and M. A. Trick. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and Welfare*, 6:157–165, 1989.
- [3] Y. Bugeaud and M. Mignotte. Polynomial root separation. *International Journal of Number Theory*, 6(3):587–602, 2010.
- [4] I. Caragiannis, C. Kaklamanis, N. Karanikolas, and A. D. Procaccia. Socially desirable approximations for Dodgson’s voting rule. *ACM Transactions on Algorithms*, 10(2):1–28, 2014.
- [5] I. Caragiannis, A. D. Procaccia, and N. Shah. When do noisy votes reveal the truth? In *Proc. of 14th EC*, pages 143–160, 2013.
- [6] G. E. Collins and A. G. Akritas. Polynomial real root isolation using Descartes’s rule of signs. In *Proc. of 3rd ISSAC*, pages 272–275, 1976.
- [7] V. Conitzer, A. Davenport, and H. Kalagnanam. Improved bounds for computing Kemeny rankings. In *Proc. of 21st AAAI*, pages 620–626, 2006.
- [8] V. Conitzer, M. Rognlie, and L. Xia. Preference functions that score rankings and maximum likelihood estimation. In *Proc. of 21st IJCAI*, pages 109–115, 2009.
- [9] V. Conitzer and T. Sandholm. Common voting rules as maximum likelihood estimators. In *Proc. of 21st UAI*, pages 145–152, 2005.
- [10] D. Coppersmith, L. Fleischer, and A. Rudra. Ordering by weighted number of wins gives a good ranking for weighted tournaments. In *Proc. of 17th SODA*, pages 776–782, 2006.
- [11] M. de Condorcet. *Essai sur l’application de l’analyse à la probabilité de décisions rendues à la pluralité de voix*. Imprimerie Royal, 1785. Facsimile published in 1972 by Chelsea Publishing Company, New York.
- [12] C. L. Dodgson. *A Method for Taking Votes on More than Two Issues*. Clarendon Press, 1876.
- [13] C. Kenyon-Mathieu and W. Schudy. How to rank with few errors. In *Proc. of 39th STOC*, pages 95–103, 2007.
- [14] E. Law and L. von Ahn. *Human Computation*. Morgan & Claypool, 2011.
- [15] G. Lebanon and J. Lafferty. Cranking: Combining rankings using conditional probability models on permutations. In *Proc. of 9th ICML*, pages 363–370, 2002.
- [16] T. Lu and C. Boutilier. Learning Mallows models with pairwise preferences. In *Proc. of 28th ICML*, pages 145–152, 2011.
- [17] K. Mahler. An inequality for the discriminant of a polynomial. *The Michigan Mathematical Journal*, 11(3):257–262, 1964.
- [18] C. L. Mallows. Non-null ranking models. *Biometrika*, 44:114–130, 1957.
- [19] A. Mao, A. D. Procaccia, and Y. Chen. Better human computation through principled voting. In *Proc. of 27th AAAI*, pages 1142–1148, 2013.
- [20] M. Meila, K. Phadnis, A. Patterson, and J. A. Bilmes. Consensus ranking under the exponential model. In *Proc. of 23rd UAI*, pages 285–294, 2007.
- [21] S. Merrill and B. Grofman. *A unified theory of voting: Directional and proximity spatial models*. Cambridge University Press, 1999.
- [22] A. D. Procaccia, S. J. Reddi, and N. Shah. A maximum likelihood approach for selecting sets of alternatives. In *Proc. of 28th UAI*, pages 695–704, 2012.
- [23] T. C. Ratliff. A comparison of Dodgson’s method and Kemeny’s rule. *Social Choice and Welfare*, 18(1):79–89, 2001.
- [24] N. Tideman. *Collective Decisions and Voting*. Ashgate, 2006.
- [25] J. V. Uspensky. *Theory of equations*. McGraw-Hill New York, 1948.
- [26] A. v. Zuylen and D. P. Williamson. Deterministic algorithms for rank aggregation and other ranking and clustering problems. In *Proc. of 5th WAOA*, pages 260–273, 2007.
- [27] L. Xia and V. Conitzer. A maximum likelihood approach towards aggregating partial orders. In *Proc. of 22nd IJCAI*, pages 446–451, 2011.
- [28] L. Xia, V. Conitzer, and J. Lang. Aggregating preferences in multi-issue domains by using maximum likelihood estimators. In *Proc. of 9th AAMAS*, pages 399–408, 2010.
- [29] H. P. Young. Condorcet’s theory of voting. *The American Political Science Review*, 82(4):1231–1244, 1988.

Iterative Splits of Quadratic Bounds for Scalable Binary Tensor Factorization

Beyza Ermis

Department of Computer Engineering
Bogazici University
34342 Bebek, Turkey
beyza.ermis@boun.edu.tr

Guillaume Bouchard

Xerox Research Centre Europe
6, Chemin de Maupertuis,
38240 Meylan, France
guillaume.bouchard@xrce.xerox.com

Abstract

Binary matrices and tensors are popular data structures that need to be efficiently approximated by low-rank representations. A standard approach is to minimize the logistic loss, well suited for binary data. In many cases, the number m of non-zero elements in the tensor is much smaller than the total number n of possible entries in the tensor. This creates a problem for large tensors because the computation of the logistic loss has a linear time complexity with n . In this work, we show that an alternative approach is to minimize the quadratic loss (root mean square error) which leads to algorithms with a training time complexity that is reduced from $O(n)$ to $O(m)$, as proposed earlier in the restricted case of alternating least-square algorithms. In addition, we propose and study a greedy algorithm that partitions the tensor into smaller tensors, each approximated by a quadratic upper bound. This technique provides a time-accuracy trade-off between a fast but approximate algorithm and an accurate but slow algorithm. We show that this technique leads to a considerable speedup in learning of real world tensors.

1 INTRODUCTION

In multi-relational data factorization problems [20, 25], many negative examples are implicitly created for relations that are *not* true. For example, in a knowledge base of family relationships, the fact `isFather(x, y)` can be considered as a positive example and automatically induces $m - 1$ negative examples of the form `not(isFather(x, z))`, for all m individuals z different from y . In other words, for some relations, one positive example is always associated with several thousands of negative examples. The focus of our work is to consider algorithms that are independent of this number of negative

examples. In a different domain, state-of-the-art detection systems in computer vision are based on a binary classifier applied many times on a dense multi-resolution scan of an image [7]. Here, most of the examples do not contain the object to be detected and negative patches often overwhelm the number of positive examples. Finally, another classical example of such problems with unbalanced categories corresponds to recommender systems taking into account implicit feedback: in this domain, it corresponds to the signal that if a user did not do some action, such as buying an object in an online shopping web site or did not click on an online advertisement, then a *negative* training example is created to take into account the fact that the proposed item or advert might not be appropriate. While these negative examples are sometimes subject to controversy since one does not know whether the recommendation was correct or not, they are nevertheless considered as very useful by practitioners and are key components of most of online recommendation engines [9].

For a binary classification problem where the total number n^+ of positive examples is largely inferior to the total number n^- of negative examples, the complexity of most of the existing learning algorithms is *at least* linear in the number $n = n^+ + n^-$ of training samples, since it is a general belief that every training point needs to be loaded in memory at least once. In fact, the sparsity of the data can be used to drastically reduce the computation time of square-norm minimization problems, as proposed by [9] using an alternating least square algorithm, where each least square problem has a complexity linear in the number of positive data only. We will give an alternative derivation of this result and show that it is also valid for gradient-based algorithms.

However, the squared loss is not always satisfactory. For example, binary tensor decomposition with logistic loss gives much better predictive performance than minimizing the squared loss. The downside of it is that the computational cost increases significantly [14, 19], and one usually relies on heuristic rules to subsample the negative examples [11]. The time to minimize the logistic loss (or

other non-quadratic loss) scales linearly in the total number $n = n^+ + n^-$ of observations, which is a real issue in these heavily unbalanced datasets for which $n^+ \ll n^-$.

In this work, the key contributions are:

- For matrix and tensor factorization models learned by minimizing the squared loss, we show that all the algorithms can benefit from this speedup, i.e. it is not restricted to the alternating least square algorithm of [8],
- We propose a new algorithm to minimize non-quadratic losses. It is based on the partitioning of the tensor into blocks and the use of Jaakkola's quadratic upper bound to the logistic loss [10]. While Jaakkola's bound has already been used to factorize large matrices, the case of unbalanced datasets was not addressed [24]. Our work can be viewed as a novel application of these upper-bounding techniques, where the incremental refinement of the approximation provides a natural way to correct the optimality gap introduced by the bound.

2 PROBLEM FORMULATION

Let $\Omega := \{(i_1, \dots, i_D) ; i_d \in \{1, \dots, n_d\} \forall d = 1, \dots, D\}$ denotes the set of D -uplets for the dimensions n_1, n_2, \dots, n_D . For each of these D -uplets, we observe a noisy binary values $y_t \in \{0, 1\}$ indexed by $t \in \Omega$. These observations can also be represented as a noisy binary tensor $Y \in \{0, 1\}^{n_1 \times \dots \times n_D}$ where n_1, n_2, \dots, n_D correspond to the tensor dimensions. Typically, $D = 2$ will correspond to binary matrices, and $D = 3$ to third-order tensors as used in database factorization models such as RESCAL [20]. Our objective is to predict the value of some specific entries in the tensor, which can be understood as detecting which entries in the tensor are outliers y_t . To do this, we estimate a tensor $Z(\theta) \in \mathbb{R}^{n_1 \times \dots \times n_D}$ of log-odds parameterized by $\theta \in \Theta$.

We formulate the problem as an empirical loss minimization. In the training phase, the empirical loss \mathcal{L} is minimized with respect to the parameter vector θ :

$$\min_{\theta \in \Theta} \mathcal{L}(\theta) \quad \mathcal{L}(\theta) := \sum_{t \in \Omega} \ell(y_t, z_t(\theta)) \quad (1)$$

where $\ell(y, z) = -y \log(\sigma(z)) - (1 - y) \log(1 - \sigma(z))$ with σ representing the sigmoid function: $\sigma(z) := \frac{1}{1 + e^{-z}}$. The predicted tensor $Z(\theta) := \{z_t(\theta)\}_{t \in \Omega}$ is assumed to be a factored representation, i.e. it has a low rank structure. For clarity, we consider only multi-linear models based on the PARAFAC [6] tensor parametrization.¹ The predictions z_t are obtained by the multilinear product of rank- K factors

¹We can easily adapt this work to more general decompositions such as Tucker or RESCAL, or even to convex loss functions using trace-norm regularization.

represented in the rows of matrices $\Theta_d \in \mathbb{R}^{n_d \times K}$, $d \in \{1, \dots, D\}$:

$$\mathbf{z}_t(\theta) := \sum_{k=1}^K \prod_{d=1}^D \theta_{t_d k}^{(d)} := \langle \theta_{t_1}^{(1)}, \dots, \theta_{t_D}^{(D)} \rangle.$$

For matrices ($D = 2$), this model is a special case of exponential-family PCA [5] where the link function is logistic. For $D = 3$, this model has been studied in the context of multi-relational knowledge bases factorization [14, 18]. To solve Equation (1), several optimization methods have been proposed in the literature. Alternating optimization, gradient descent and stochastic gradient descent:

- The gradient descent algorithms are only based on the minimization of \mathcal{L} by doing small steps in the direction of the gradients. Since the loss is differentiable, we derive its gradient in closed form and use a generic software to choose the optimal descent direction. In the experiments below, we use Marc Schmidt's `minFunc` Matlab function.²
- The alternating optimization procedure, also called block coordinate descent, consists in minimizing the loss \mathcal{L} with respect to the i -th component Θ_i , keeping all the other components Θ_j , $j \neq i$ fixed. This optimization makes use of existing optimized linear logistic regression algorithms. The optimization procedure is obtained through a round-robin schedule. This approach is simple to implement, but it involves an inner loop since linear logistic regression algorithms are also based on gradient minimization.
- For large scale optimization, there is a growing interest in stochastic gradient descent algorithms since every gradient computation can potentially be too expensive. It consists in computing the gradients for a subset of the observations.

For highly sparse matrices where the number of zeros is much larger than the number of ones, these approaches do not scale well with the dimension of the tensor. For each of these algorithms, the time to make one function evaluation is the key bottleneck. The gradient descent algorithm requires to sum $|\Omega|$ elements (one per possible prediction). One step of the alternating optimization procedure is computationally costly because it requires to solve a linear logistic regression with $|\Omega|$ observations. The stochastic gradient descent algorithm seems to be better as each iteration is very fast, but it still requires $|\Omega|$ iterations to do one pass through the data. For some problems, good predictive performances are obtained even before the first pass

²Can be found at <http://www.di.ens.fr/~mschmidt/Software/minFunc.html>.

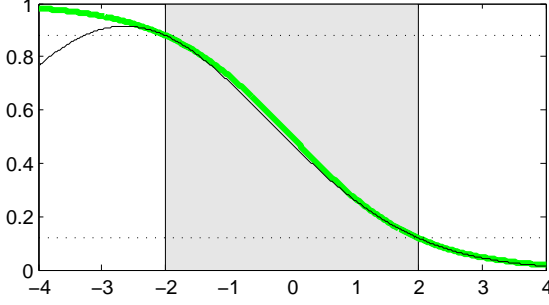


Figure 1: Comparison of the logistic function and the Gaussian distribution. The Gaussian distribution (the mean is -2.61 and the variance is 5.25) approximates well the logistic function in a range of values (gray area in the range [-2,2]) that corresponds to probabilities between 0.12 and 0.88 (dotted lines). In most of recommender systems applications, the probability of correctly predicting user choice is in this range, which explains that RMSE loss is reasonable.

through the data has been completed, but this case is relatively rare in practice. Our objective is to study methods that have a sub-linear complexity in the number of training sample, i.e. we can take into account *all* the $|\Omega|$ training samples while having a complexity that scales only in the number $n^+ = |\Omega^+|$ of non-zero elements. In the following we show that this complexity can be obtained by using a quadratic approximation to the logistic loss, leading to considerable speedup in our experiments.

3 QUADRATIC LOSS: FAST BUT OFTEN INACCURATE

As illustrated in Figure 1, the logistic loss can be reasonably approximated by a quadratic function, so the Root Mean Square Error (RMSE) should be a good surrogate function to minimize. A naive computation of the square loss $\mathcal{L}(\theta) = \sum_{t \in \Omega} (y_t - z_t(\theta))^2$ would require $O(KD \prod_d n_d)$ operations, since there are $\prod_d n_d$ possible predictions, but simple algebra shows that it is equal to:

$$\mathcal{L}(\theta) = \sum_{t \in \Omega} (y_t - z_t)^2 \quad (2)$$

$$= n^+ - 2 \sum_{t \in \Omega^+} z_t(\theta) + \sum_{k=1}^K \sum_{k'=1}^K \prod_{d=1}^D M_{kk'}^{(d)} \quad (3)$$

where the $K \times K$ matrices $M^{(d)}$ are defined by $M_{kk'}^{(d)} := \sum_{j=1}^d \theta_{jk}^{(d)} \theta_{jk'}^{(d)}$. Hence, for tensors of high dimension, we get a significant speedup as Equation (3) can be computed in $O(Kn^+ + K^2 \sum_d n_d)$ operations. As an example, assume one wishes to compute the loss of a $1000 \times 1000 \times 1000$ tensor containing 10^5 entries and the low-rank approximation has rank $K = 100$. Then, we can see that there are 3.10^{11} basic operations in the formula of Equation (2),

while the formula of Equation (3) contains 6.10^7 basic operations. This means that it will be 5000 times faster to compute exactly the same quantity!

If we minimize the loss $\mathcal{L}(\theta)$ with respect to θ using block-coordinate descent, the iterations end up being least squares problem with a per-iteration complexity that scales linearly with the number of positive examples only. This corresponds exactly to the iTALS algorithm [22], which is the tensor generalization of the alternating least squares algorithm of [9]. In our experiments, we used gradient descent to minimize the objective function, using Equation (3) to compute the gradient efficiently (the complexity is the same as the function evaluation).

4 SPEED-ACCURACY TRADE-OFF BY BOUNDING SPLITS

4.1 UPPER BOUNDING THE LOSS

To speed-up computation, we minimize a quadratic upper bound to the logistic loss. We use Jaakkola's bound to the logistic loss [10]:

$$\log(1 + e^z) \leq \lambda(\xi)(z^2 - \xi^2) + \frac{1}{2}(z - \xi) + \log(1 + e^\xi),$$

where $\lambda(\xi) := \frac{1}{2\xi}(\frac{1}{1+e^\xi} - \frac{1}{2})$ and ξ is a variational parameter. We keep the same value for ξ for all the elements of the tensor Z , so that the upper bound has exactly the form required to apply the computational speedup described in the previous section.

$$\bar{\mathcal{L}}(\theta, \xi) = \lambda(\xi) \sum_{t \in \Omega} \left(z_t - \frac{2y_t}{4\lambda(\xi)} \right)^2 + c(\xi)$$

where $c(\xi)$ is a constant function that does not depend on θ :

$$c(\xi) = |\Omega| \left(\log(1 + e^\xi) - \lambda(\xi)\xi^2 - \frac{1}{2}\xi - \frac{1}{16\lambda(\xi)} \right)$$

The optimization of this bound with respect to ξ gives exactly the Frobenius norm of the tensor:

$$\sum_{t \in \Omega} z_t(\theta)^2 = \arg \min_{\xi} \bar{\mathcal{L}}(\theta, \xi). \quad (4)$$

Note that $Z(\theta)$ is low rank in general. This means that it can also be efficiently computed using the third term of Equation (3). We have now an upper bound to the original loss \mathcal{L} that needs to be minimized:

$$\mathcal{L}(\theta) \leq \bar{\mathcal{L}}^\Omega(\theta, \xi). \quad (5)$$

As usual with bound optimization, we alternate between two steps: 1) minimizing the bound with respect to the variational parameter ξ using closed form updates (e.g. when using Jaakkola's bound) or dichotomic search where

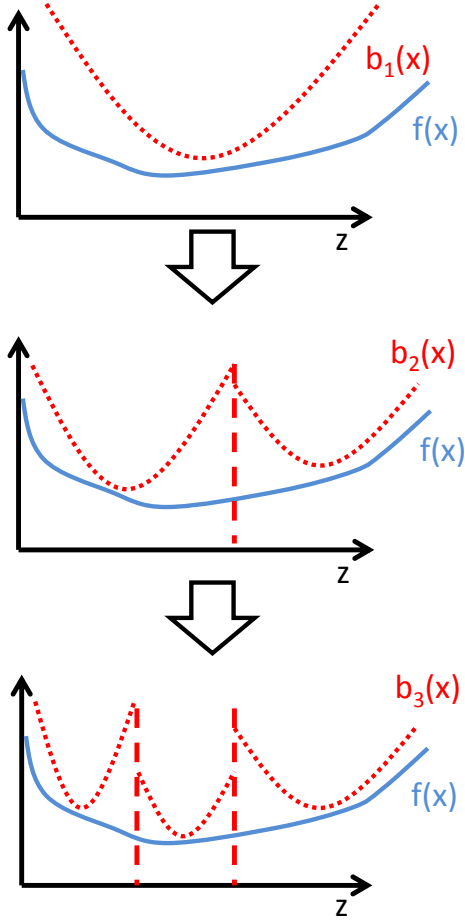


Figure 2: Illustration of the idea of bound refinement. The main idea is that computing the integral of the quadratic upper bound (such as b_1 in the top graph) is much faster than computing the integral of f directly. To improve the accuracy, we use piecewise bounds. To choose the domain of the pieces, we use a greedy algorithm that identifies the partition that leads to the diminished upper bound (leading to the upper bounds b_2 and b_3).

no closed form solution exists; and 2) minimizing the bound with respect to θ using a standard gradient descent algorithm. This algorithm is sometimes referred as Majorization-Minimization algorithm in the literature [16]. The algorithm minimizes the loss with a complexity per iteration equal to $O(Kn^+ + K^2 \sum_d n_d)$. In the experiments, this algorithm is called *Quad-App*. It is detailed in Algorithm 2.

4.2 SPLIT THE DATA TO IMPROVE ACCURACY

The drawback of the previous approach, even with one or two orders of magnitude speedups, the resulting quadratic approximation can be quite loose for some data, and the accuracy of the method can be too low. We give here a family of approximation that interpolates between this fast

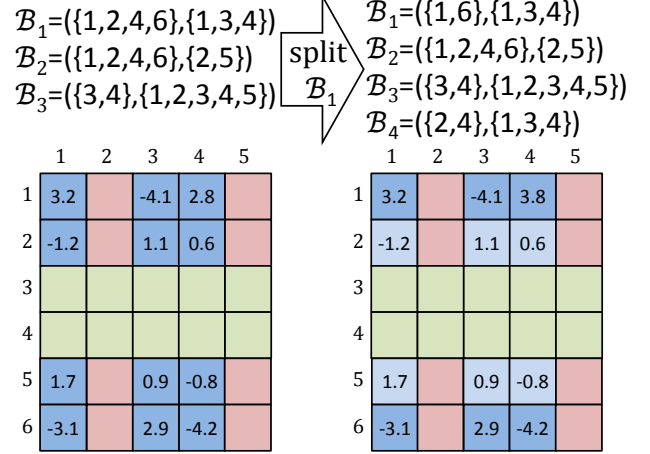


Figure 3: Example of matrix split to improve upper bound accuracy. Colors identify blocks. Numbers in the matrix represent predictions. The block 1 is decomposed into blocks 1 and 4, both having a small variance on the absolute value of the predictions.

but inaccurate quadratic approximation and the slow but exact minimization of the non-quadratic loss.

We propose to take advantage of the speedup due to the quadratic upper bound by applying it on a partition of the original tensor: we select a set $\mathcal{B} = \{B_1, \dots, B_{|\mathcal{B}|}\}$ of disjoint *blocks* that partitions the space of possible observation indices Ω . On each of these blocks, the bounding technique described in the previous section is applied, the main difference being that the minimization with respect to θ is done jointly on all the blocks. This process is illustrated in Figure 2. Formally, each block B_b , $b \in \{1, \dots, |\mathcal{B}|\}$ is identified by D sets of indices which represent the dimensions that are selected in the given block b . The split of these indices are illustrated in a toy matrix example in Figure 3. Blocks for tensors are computed the same way as matrices, but the depth indices are also splitted: At each refinement step, we choose to partition the rows indices, column indices or depth indices.

To select the blocks, we use a greedy construction of the blocks: starting with a single block containing all the indices, i.e. $\mathcal{B} = \{\Omega\}$, we iteratively refine the blocks using the following two-step procedure, called *RefineBlocks*:

1. select the block b to split that has the maximal variance in the absolute values of the predictions $|z_t|$;
2. split the block b into two block so that the variance of the absolute values of the predictions $|z_t|$ is minimized.

An example of such a split is shown in Figure 3. This method is fully described in Algorithm 1, which uses Al-

Algorithm 1 Iterative block splitting: $\min_{\theta, \mathcal{B}} \bar{\mathcal{L}}(\theta, \mathcal{B}, Y)$

- 1: $\hat{\theta} = \text{UBAdaptiveMinimization}(Y, \theta^{(0)}, \varepsilon)$
 - 2: **Inputs:** tensor Y , initial $\theta^{(0)}$, tolerance ε
 - 3: **Outputs:** latent factors $\hat{\theta} = \{\hat{\theta}_1, \dots, \hat{\theta}_D\}$
 - 4: Initialize blocks $\mathcal{B}^{(0)} = \{\Omega\}$,
 - 5: **for** $i = 1, 2, \dots$ until improvement less than ε **do**
 - 6: $\theta^{(i)} = \text{UBMinimization}(Y, \theta^{(i-1)}, \varepsilon/2, \mathcal{B}^{(i-1)})$
 - 7: $\mathcal{B}^{(i)} \leftarrow \text{RefineBlocks}(\mathcal{B}^{(i-1)}, \theta^{(i)})$
 - 8: **end for**
 - 9: $\hat{\theta} = \theta^{(i)}$
-

Algorithm 2 $\text{UBMinimization} \min_{\theta} \bar{\mathcal{L}}(\theta, \mathcal{B}, Y)$

- 1: $\hat{\theta} = \text{UBMinimization}(Y, \theta^{(0)}, \varepsilon, \mathcal{B})$
 - 2: **Inputs:** tensor Y , initial $\theta^{(0)}$, tol. ε , blocks \mathcal{B}
 - 3: **Outputs:** latent factors $\hat{\theta} = \{\hat{\theta}_1, \dots, \hat{\theta}_D\}$
 - 4: **for** $i = 1, 2, \dots$ until improvement less than ε **do**
 - 5: **for** $b = 1, 2, \dots, |\mathcal{B}|$ **do**
 - 6: $\xi_b^{(i)} \leftarrow \arg \min_{\xi_b} \bar{\mathcal{L}}_b(\theta^{(i-1)}, \xi_b, \mathcal{B}_b)$
 - 7: **end for**
 - 8: $\theta^{(i)} \leftarrow \arg \min_{\theta} \sum_b \bar{\mathcal{L}}_b(\theta, \xi_b^{(i)}, \mathcal{B}_b)$
 - 9: **end for**
 - 10: $\hat{\theta} = \theta^{(i)}$
-

gorithm 2 as a sub-program to learn the parameters for a fixed set of blocks, following the approach described in the previous section. There is a tradeoff between optimizing the bound using Algorithm 2 and refining the partition \mathcal{B} in Algorithm 1. A simple strategy that worked well in practice was to refine the partition when the upper bound minimization did not improve more than a given tolerance level $\varepsilon/2$ in two successive iterations. This piecewise refinement strategy is called *PW Quad-App* in the experimental section.

We also introduced a slight variant of this basic algorithm, where we compute the exact logistic loss in blocks that are sufficiently dense, i.e. when computing Equation (3) requires more iterations than computing the loss in the classical way (Equation (3)). This condition is verified when $n^+(B) \geq \prod_{d=1}^D n_d^{(B)} - K \sum_{d=1}^D n_d^{(B)}$, where $n^+(B)$ and $n_d^{(B)}$ correspond to the number of non-zero elements and dimensions of the block tensor B . We call this variant *PW Quad-App + Logistic*.

5 EXPERIMENTS AND RESULTS

In this section, to evaluate the performances of our framework, we conducted experiments on both synthetic and real datasets.

Synthetic Data Experiments To explore the speed-accuracy tradeoff, we generated different binary matrices Y by randomly sampling noisy low-rank matrices $X =$

$UV + E$ where $U \in \mathbb{R}^{n_1 \times r}$ and $V \in \mathbb{R}^{r \times n_2}$ are generated using independent standard normal variables and $E \in \mathbb{R}^{n_1 \times n_2}$ is a normally distributed Gaussian noise with standard deviation σ . To create the binary matrix $Y \in \{0, 1\}^{n_1 \times n_2}$, we round the values of X using a high percentile of X as a threshold to produce a heavy tendency towards the negative class. We learn an estimation \hat{X} of the original matrix X on the data matrix Y assumed to be fully observed and compute the RMSE on the recovery of X , i.e. $\text{RMSE} = \|X - \hat{X}\|_F$. We measure the running time of each of the methods to understand their scalability to large datasets.

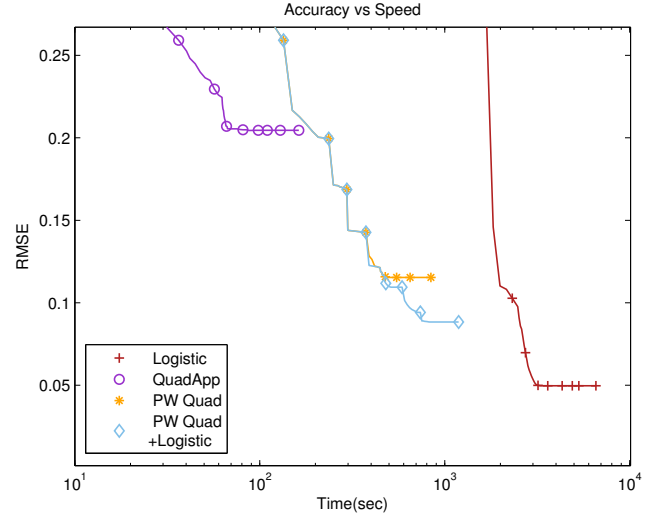


Figure 4: Matrix recovery results on simulation data with size 10000×5000 , sparsity %0.1 and noise $\sigma = 0.1$. Markers are plotted at iterations 10, 17, 28, 35, 52, 63 and 73 (these times correspond to the refinement of the piecewise bound).

The timing and accuracy performances of our methods on simulation data with various dimensions, noise levels and sparsity percentages are shown in Table 1. These results are averaged over 10 runs and we choose rank $r = 5$ for every simulation. Here, the baselines are logistic loss and quadratic loss. The logistic loss gives much smaller error rates than the quadratic loss (EUC-Full and EUC-Fast) and quadratic approximation, especially when the noise level is low. However, minimizing the logistic loss requires considerably more time than the alternative approaches as the problem size grows. These experiments highlight the fact that our unified framework for quadratic loss gives a significant improvement over logistic loss in terms of runtime performance. We also observe that the piecewise quadratic bounding technique has better predictive performance than the quadratic approximation, along with a huge speedup when we compare it to the time to train the model using the logistic loss. On Figure 4, we plotted the test RMSE with respect to the CPU time (each marker corresponds to

Table 1: Evaluation results of the synthetic experiments in terms of seconds for runtime and RMSE for matrix recovery.

Noise Level	Dimension	Sparsity	Methods									
			EUC-Full		EUC-Fast		Logistic		Quad-App		PW QuadApp	
			RMSE	Time	RMSE	Time	RMSE	Time	RMSE	Time	RMSE	Time
Low Noise $\sigma = 0.1$	$n_1 = 100$ $n_2 = 50$	10%	0.6970	60.45	0.6970	0.50	0.3561	103.13	0.6421	0.58	0.4377	6.16
		1%	0.6792	55.57	0.6792	0.46	0.1095	90.65	0.4568	0.56	0.1918	3.19
	$n_1 = 1000$ $n_2 = 500$	1%	0.7251	5295.7	0.7251	63.10	0.1563	7216.2	0.7054	75.49	0.3790	421.11
		0.1%	0.7251	5248.3	0.7251	42.90	0.2126	6605.6	0.7067	62.90	0.5247	301.35
	$n_1 = 10000$ $n_2 = 5000$	0.1%	0.4483	84950	0.4483	1289.2	0.0497	68199	0.2052	1353.8	0.0911	6683.9
		0.01%	0.4217	86109	0.4217	803.2	0.0329	66482	0.1814	1049.3	0.0583	4271.4
High Noise $\sigma = 2.0$	$n_1 = 100$ $n_2 = 50$	10%	2.8989	59.06	2.8989	0.48	1.2789	94.49	1.8639	0.73	1.3212	10.18
		1%	2.8821	44.96	2.8821	0.37	0.2377	59.21	0.4589	0.54	0.2622	7.14
	$n_1 = 1000$ $n_2 = 500$	1%	2.6705	7070.3	2.6705	110.87	0.3371	6592.0	0.4052	132.55	0.3659	943.67
		0.1%	2.5116	7276.2	2.5116	109.99	0.0563	6503.1	0.1304	120.83	0.1078	783.91
	$n_1 = 10000$ $n_2 = 5000$	0.1%	1.9990	97084	1.9990	1486.1	0.2312	66124	0.2604	1523.0	0.2374	7352.8
		0.01%	1.7416	83846	1.7416	1183.0	0.1332	65664	0.1661	1589.6	0.1414	6613.1

an iteration). Because the error rate of quadratic loss is considerably bigger than the other methods, we show the results of logistic loss, quadratic approximation and piecewise methods in this figure. It is interesting to see that the piecewise quadratic bound reaches its goal of interpolating the performances between the fast but inaccurate quadratic approximation, and the slow but accurate logistic loss minimization: On a wide range of times (from 2 minutes to 1 hour), the piecewise quadratic bound gives the best performances. It is worth to note that the slight modification that was introduced to perform exact logistic on the smallest pieces improves performances when the methods have nearly converged (after 60 iterations). Note that this experiment was done on matrices, but the differences are even bigger for tensors of high order.

Real Data Experiments In order to evaluate the performances of our methods, we designed link-prediction experiments on standard multi-relational datasets: *Nations* that groups 14 countries (entities) with 56 binary relation types (like 'economic aid', 'treaties' or 'rel diplomacy') representing interactions among them; *Kinships* which is the complex relational structure of Australian tribes' kinship systems. In *Kinships* dataset, 104 tribe members were asked to provide the kinship terms they used for one another and this results in graph of 104 entities and 26 relation types, each of them depicting a different kinship term. And *UMLS* that contains data from the Unified Medical Language System semantic work used in [11]. This dataset consists in a graph with 135 entities (high-level concepts like 'Disease or Syndrome', 'Diagnostic Procedure') and 49 relation types (verbs depicting causal influence between concepts like 'affect' or 'cause'). In the end, these datasets results in tensors $Y \in \{0, 1\}^{14 \times 14 \times 56}$, $Y \in \{0, 1\}^{104 \times 104 \times 26}$ and $Y \in \{0, 1\}^{135 \times 135 \times 49}$ respec-

tively.

Then, we compared the Area Under the Receiver Operating Characteristic Curve (AUC) and runtime in seconds of piecewise methods to the results of quadratic approximation and logistic loss and also the results of RESCAL [20], SME [2] and LFM [11] that have the best published results on these benchmarks in terms of AUC.

In addition, we test the performances of these methods on three datasets in matrix form: MovieLens ³, Last FM ⁴ [4] and Sushi Preference [12]. MovieLens dataset contains movie ratings of approximately 1682 movies made by 943 MovieLens users and results in matrix $Y \in \{0, 1\}^{943 \times 1682}$. The Last FM dataset consists of music artist listening information from a set of 1892 users from *Last.fm* online music system. We construct a binary matrix $Y \in \{0, 1\}^{1892 \times 17632}$ from this dataset that contains the artists listened by each user. Lastly, the *Sushi Preference Data Set* includes 4950 users' responses of preference in 100 different kinds of *sushi*. In this dataset, the most disliked kind of sushi represented by 0 and the most preferred one is represented by 1. Eventually, sushi dataset results in matrix $Y \in \{0, 1\}^{100 \times 4950}$.

For the results given in Table 2, we performed 10-fold cross validation and averaged over 10 random splits of the datasets. In addition, we select the optimal regularization parameter λ^* by searching over the set $\{0.01, 0.05, 0.1, 0.5, 1\}$ that maximizes the AUC and we computed rank-20 decomposition of these datasets in order to get comparable results to RESCAL, SME and LFM. The time and accuracy comparisons are given in Table 2 in terms of seconds for time and AUC metric for predict-

³www.grouplens.org/node/73

⁴www.grouplens.org/datasets/hetrec-2011/

Table 2: Evaluation results obtained by our approaches, RESCAL [20], SME [2] and LFM [11] on the given datasets.

	Methods	Datasets					
		Nations	Kinships	UMLS	MovieLens	Last FM	Sushi
AUC	EUC-Full	0.7536	0.8193	0.8205	0.8511	0.8965	0.8199
	EUC-Fast	0.7536	0.8193	0.8205	0.8511	0.8965	0.8199
	Logistic	0.9253	0.9592	0.9795	0.9848	0.9454	0.9513
	Quad-App	0.8635	0.9087	0.9169	0.8916	0.9042	0.9078
	PW QuadApp	0.9038	0.9213	0.9387	0.9490	0.9272	0.9200
	PW Quad+Logistic	0.9122	0.9416	0.9566	0.9781	0.9381	0.9373
	RESCAL [20]	0.8400	0.9500	0.9800	0.9601	0.9257	0.9481
	SME [2]	0.8830	0.9070	0.9830	0.9144	0.9328	0.9177
Time (sec)	LFM [11]	0.9090	0.9460	0.9900	0.9790	0.9401	0.9598
	EUC-Full	922.87	8793.8	81039	103454	701819	13490.8
	EUC-Fast	18.37	80.95	167.67	344.81	3688.74	142.63
	Logistic	1483.7	10374.2	75489	111257	721994	12704.6
	Quad-App	18.07	97.13	187.1	431.35	1839.16	142.06
	PieceQuadApp	32.52	169.35	922.65	1095.94	2792.18	643.56
	PW Quad+Logistic	59.71	651.12	1035.47	1349.6	4065.12	755.72
	RESCAL [20]	626.40	3714.6	4142.05	6786.23	9861.50	2632.1
	SME [2]	32.11	135.9	513.03	749.07	1627.56	279.38
	LFM [11]	64.63	1446.22	5097.5	8265.56	13058.1	3438.07

ing missing values. These results demonstrate that logistic loss improves accuracy over RESCAL, SME and LFM in Nations, Kinships, MovieLens and Last FM datasets while it reaches almost the same score for UMLS and Sushi datasets. On the other side, piecewise methods provide very close approximation to logistic loss on all these datasets and they have a significant advantage in terms of runtime over the other methods. They take a small fraction of logistic loss’ running time, especially for large datasets.

6 RELATED WORK

In order to deal with learning on various forms of structured data such as large-scale knowledge bases, time-varying networks or recommendation data, tensor factorizations have become increasingly popular [3, 21, 23]. Recently, Nickel et al presented RESCAL [20], an upgrade over previous tensor factorization methods, which has been shown to achieve state-of-the-art results for various relational learning tasks such as link prediction and entity resolution. Independently, a similar logistic extension of the RESCAL factorization has been proposed in [14]. [21] is an extension to the RESCAL algorithm on the YAGO ontology and is based on alternating least-squares updates of the factor matrices, has been shown to scale up to large knowledge bases via exploiting the sparsity of relational data. Among the existing works, [18] is the most similar work with our, which is the logistic extension of RESCAL. It demonstrates that the logistic loss improves the prediction results significantly but their algorithm requires to compute the dense matrix and cannot scale to large data. In RESCAL, entities are modeled by real-valued vectors and relations by matri-

ces. Bordes *et al* has further improved this idea in the Structured Embeddings (SE) framework [3] by learning a model to represent elements of any knowledge base (KB) into a relatively low dimensional embedding vector space by tensor factorization method. Latent Factor Model (LFM) [11] is based on a bilinear structure, which captures various orders of interaction of the data, and also shares sparse latent factors across different relations. In [2], they present a new neural network designed to embed multi-relational graphs into a flexible continuous vector space via a custom energy function (SME) in which the original data is kept and enhanced. In all of these studies [3, 11, 2], the data is extremely skewed i.e., the number of negative examples \gg the number of positive examples. To overcome the sparsity, they first select a positive training triplet at random, then create a negative triplet by sampling an entity from the set of all entities. Unlike these approaches, we argue that it is in general more appropriate to consider *all* the negative examples.

Maaten et al [15] derive an upper bound to logistic loss which can be minimized as surrogate loss for linear predictors on binary labels. Khan et al [13] used Jaakkola’s bound for binary observations and Bohning’s bound for multinomial observations [1]. Our work can be easily extended to take into account Bohning’s bound. In addition, piecewise bounds have the important property: reducing the error as the number of pieces increase. Marlin et al. proposed an improvement on the logistic-loss with piecewise linear bounds, but this is a local approach and does not apply in our setting since we need a global quadratic bound to apply the squared norm trick [17].

7 CONCLUSION

There were several important techniques used in this paper: 1) the decomposition of the loss into a small positive part and a large but structured negative space; 2) the use of the squared norm trick that reduces the complexity of squared loss computation and 3) the use of the partitioning technique to gradually reduce the gap introduced by the usage of quadratic upper bounds for non-quadratic losses, particularly useful in the case of binary or count data. This combination of techniques can be applied in a broad range of other problems, such as probabilistic CCA, collective-matrix factorization, non-negative matrix factorization, as well as non-factorial models such as time series [7].

ACKNOWLEDGEMENTS

This work has been supported by the Fupol and Fusepool FP7 European project. We thank anonymous reviewers, Jean-Marc Andreoli and Jos-Antonio Rodriguez for their comments.

References

- [1] D. Bohning. Multinomial logistic regression algorithm. *Annals of the Inst. of Statistical Math.*, 44:197200, 1992.
- [2] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. A semantic matching energy function for learning with multi-relational data. *Machine Learning: Special Issue on Learning Semantics*, 2013.
- [3] Antoine Bordes, Jason Weston, Ronan Collobert, and Yoshua Bengio. Learning structured embeddings of knowledge bases. In *AAAI*, 2011.
- [4] Iván Cantador, Peter Brusilovsky, and Tsvi Kuflik. 2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In *Proceedings of the 5th ACM conference on Recommender systems*, RecSys 2011, New York, NY, USA, 2011. ACM.
- [5] Michael Collins, Sanjoy Dasgupta, and Robert E Schapire. A generalization of principal component analysis to the exponential family. In *NIPS*, volume 13, page 23, 2001.
- [6] Richard A Harshman and Margaret E Lundy. The PARAFAC model for three-way factor analysis and multi-dimensional scaling. *Research methods for multimode data analysis*, pages 122–215, 1984.
- [7] Joo F. Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *ECCV (4)*, pages 702–715, 2012.
- [8] Balzs Hidasi and Domonkos Tikk. Fast ALS-Based tensor factorization for context-aware recommendation from implicit feedback. *ECML PKDD*, page 6782, Bristol, UK, 2012.
- [9] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *ICDM*, page 263272, 2008.
- [10] T Jaakkola and M Jordan. A variational approach to Bayesian logistic regression models and their extensions. In *Sixth International Workshop on Artificial Intelligence and Statistics*, 1997.
- [11] Rodolphe Jenatton, Nicolas Le Roux, Antoine Bordes, and Guillaume Obozinski. A latent factor model for highly multi-relational data. In *NIPS*, pages 3176–3184, 2012.
- [12] Toshihiro Kamishima. Nantonac collaborative filtering: Recommendation based on order responses. In *Proceedings of the 9th International Conference on Knowledge Discovery and Data Mining*, page 583588. ACM, August 2003.
- [13] Mohammad Emtiyaz Khan, Benjamin M. Marlin, Guillaume Bouchard, and Kevin P. Murphy. Variational bounds for mixed-data factor analysis. In *NIPS*, pages 1108–1116, 2010.
- [14] Ben London, Theodoros Rekatsinas, Bert Huang, and Lise Getoor. Multi-relational learning using weighted tensor decomposition with modular loss. *arXiv preprint arXiv:1303.1733*, 2013.
- [15] Laurens Maaten, Minmin Chen, Stephen Tyree, and Kilian Q Weinberger. Learning with marginalized corrupted features. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, page 410418, 2013.
- [16] Julien Mairal. Optimization with first-order surrogate functions. In *ICML*, 2013.
- [17] Benjamin M. Marlin, Mohammad Emtiyaz Khan, and Kevin P. Murphy. Piecewise Bounds for Estimating Bernoulli-Logistic Latent Gaussian Models. In *ICML*, pages 633–640, 2011.
- [18] Maximilian Nickel and Volker Tresp. Logistic tensor factorization for multi-relational data. *CoRR*, abs/1306.2084, 2013.
- [19] Maximilian Nickel and Volker Tresp. Tensor factorization for multi-relational learning. In *Machine Learning and Knowledge Discovery in Databases*, page 617621. Springer, 2013.
- [20] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *ICML*, page 809816, 2011.
- [21] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. Factorizing YAGO: scalable machine learning for linked data. In *WWW*, page 271280, Lyon, France, 2012.
- [22] Istvn Pilszy, Dvid Zibriczky, and Domonkos Tikk. Fast ALS-based matrix factorization for explicit and implicit feedback datasets. In *ACM RecSys*, page 7178, 2010.
- [23] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized Markov chains for next-basket recommendation. In *WWW*, page 811820, 2010.
- [24] Matthias Seeger and Guillaume Bouchard. Fast Variational Bayesian Inference for Non-Conjugate Matrix Factorization Models. *Journal of Machine Learning Research - Proceedings Track*, 22:1012–1018, 2012.
- [25] Ajit P. Singh and Geoffrey J. Gordon. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, page 650658. ACM, New York, NY, USA, 2008.

Finding Optimal Bayesian Network Structures with Constraints Learned from Data

Xiannian Fan¹, Brandon Malone² and Changhe Yuan¹

¹Queens College/City University of New York

²Helsinki Institute for Information Technology,

xfan2@qc.cuny.edu, brandon.malone@cs.helsinki.fi, changhe.yuan@qc.cuny.edu

Abstract

Several recent algorithms for learning Bayesian network structures first calculate potentially optimal parent sets (POPS) for all variables and then use various optimization techniques to find a set of POPS, one for each variable, that constitutes an optimal network structure. This paper makes the observation that there is useful information implicit in the POPS. Specifically, the POPS of a variable constrain its parent candidates. Moreover, the parent candidates of all variables together give a directed cyclic graph, which often decomposes into a set of strongly connected components (SCCs). Each SCC corresponds to a smaller subproblem which can be solved independently of the others. Our results show that solving the constrained subproblems significantly improves the efficiency and scalability of heuristic search-based structure learning algorithms. Further, we show that by considering only the top p POPS of each variable, we quickly find provably very high quality networks for large datasets.

1 INTRODUCTION

Bayesian networks (BNs) are graphical models that represent uncertain relationships between random variables. While BNs have become one of the most popular and well-studied probabilistic model classes, a common bottleneck lies in deciding upon their structure. Often, experts are unable to completely specify the structure; in these cases, a good structure must be learned from expert knowledge and available data. In this work, we consider the problem of exact, score-based Bayesian network structure learning (BNSL), which is known to be NP-hard (Chickering 1996).

Despite the difficulty of BNSL, though, a variety of algorithms have been proposed which can solve modest-sized learning problems. The first exact algorithms were

based on dynamic programming (Koivisto and Sood 2004; Ott, Imoto, and Miyano 2004; Singh and Moore 2005; Silander and Myllymäki 2006). Later algorithms have used strategies such as integer linear programming (Jaakkola et al. 2010; Cussens 2011; Bartlett and Cussens 2013) and heuristic search (Yuan and Malone 2013; Malone et al. 2011; Malone and Yuan 2013). These algorithms generally take as input the potentially optimal parent sets (POPS) for each variable. They all improve upon dynamic programming by, either implicitly or explicitly, pruning the search space and considering only promising structures.

In this paper, we focus on the heuristic search approach first proposed by Yuan *et al.* (2011) in which BNSL is formulated as a shortest-path finding problem. A state space search strategy like A* or breadth-first branch and bound is then used to solve the transformed problem. Previous work in heuristic search for BNSL has focused on pruning unpromising structures based on bounds derived from admissible heuristic functions. In this work, we show that *POPS constraints*, which are implicit in the problem input, significantly improve the efficiency of the search by pruning large portions of the search space.

The remainder of this paper is structured as follows. Section 2 provides an overview of BNSL and the shortest-path finding formulation of the problem. Section 3 introduces POPS constraints and shows how they can be used to prune the search space. Additionally, we describe a pruning strategy which uses the constraints to trade guaranteed bounded optimality for more scalable performance in Section 4. The POPS constraints also reduce the space required by the heuristics used for pruning during search, as described in Section 5. In Section 6, we compare POPS constraints to related work. Section 7 gives empirical results on a set of benchmark datasets, and Section 8 concludes the paper.

2 BACKGROUND

This section reviews BNSL and the shortest-path finding formulation of the learning problem (Yuan and Malone 2013), which is the basis of our new algorithm.

2.1 BAYESIAN NETWORK STRUCTURE LEARNING

A Bayesian network (BN) consists of a directed acyclic graph (DAG) in which the vertices correspond to a set of random variables $\mathbf{V} = \{X_1, \dots, X_n\}$ and a set of conditional probability distributions $P(X_i|PA_i)$, where all parents of X_i are referred to as PA_i . The joint probability over all variables factorizes as the product of the conditional probability distributions.

We consider the problem of learning a network structure from a discrete dataset $\mathbf{D} = \{D_1, \dots, D_N\}$, where D_i is an instantiation of all the variables in \mathbf{V} . A scoring function s measures the goodness of fit of a network structure to \mathbf{D} (Heckerman 1998). The goal is to find a structure which optimizes the score. We only require that the scoring function is *decomposable* (Heckerman 1998); that is, the score of a network $s(N) = \sum_i s_i(PA_i)$. The $s_i(PA_i)$ values are often called *local scores*. Many commonly used scoring functions, such as MDL (Lam and Bacchus 1994) and BDe (Buntine 1991; Heckerman, Geiger, and Chickering 1995), are decomposable.

2.2 LOCAL SCORES

While the local scores are defined for all 2^{n-1} possible parent sets for each variable, this number is greatly reduced by pruning parent sets that are provably never optimal (de Campos and Ji 2011). We refer to this as *lossless score pruning* because it is guaranteed to not remove the optimal network from consideration. We refer to the scores remaining after pruning as *potentially optimal parent sets* (POPS).

Other pruning strategies, such as restricting the cardinality of parent sets, are also possible, but these techniques could eliminate parent sets which are in the globally optimal network; we refer to pruning strategies which might remove the optimal network from consideration as *lossy score pruning*. Of course, these, and any other, score pruning strategies can be combined.

Regardless of the score pruning strategies used, we still refer to the set of unpruned local scores as POPS and denote the set of POPS for X_i as \mathcal{P}_i . The POPS are given as input to the learning problem. We define the *Bayesian network structure learning problem* (BNSL) as follows.

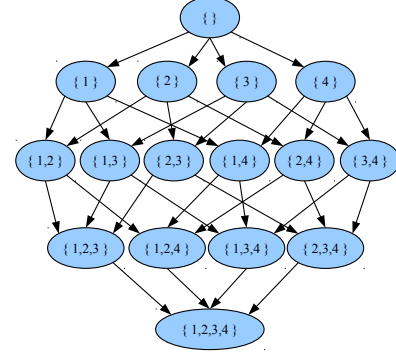


Figure 1: An order graph for four variables.

The BNSL Problem

INPUT: A set $\mathbf{V} = \{X_1, \dots, X_n\}$ of variables and a set of POPS \mathcal{P}_i for each X_i .

TASK: Find a DAG N^* such that

$$N^* \in \arg \min_N \sum_{i=1}^n s_i(PA_i),$$

where PA_i is the parent set of X_i in N and $PA_i \in \mathcal{P}_i$.

2.3 SHORTEST-PATH FINDING FORMULATION

Yuan and Malone (2013) formulated BNSL as a shortest-path finding problem. Figure 1 shows the *implicit* search graph for four variables. The top-most node with the empty variable set is the *start* node, and the bottom-most node with the complete set is the *goal* node. An arc from \mathbf{U} to $\mathbf{U} \cup \{X_i\}$ in the graph represents generating a successor node by adding a new variable X_i as a leaf to an existing subnetwork of variables \mathbf{U} ; the cost of the arc is equal to the score of the optimal parent set for X_i out of \mathbf{U} , which is computed by considering all subsets of the variables in $PA \subseteq \mathbf{U}, PA \in \mathcal{P}_i$, i.e.,

$$\begin{aligned} \text{cost}(\mathbf{U} \rightarrow \mathbf{U} \cup \{X_i\}) &= \text{BestScore}(X_i, \mathbf{U}) \\ &= \min_{PA_i \subseteq \mathbf{U}, PA_i \in \mathcal{P}_i} s_i(PA_i). \end{aligned} \quad (1) \quad (2)$$

In this search graph, each path from *start* to *goal* corresponds to an ordering of the variables in the order of their appearance, so the search graph is also called as *order graph*. Each variable selects optimal parents from the variables that precede it, so combining the optimal parent sets yields an optimal structure for that ordering. The shortest path gives the global optimal structure.

2.4 HEURISTIC SEARCH ALGORITHMS

This shortest path problem has been solved using several heuristic search algorithms, including A* (Yuan, Malone, and Wu 2011), anytime window A* (AWA*) (Malone and Yuan 2013) and breadth-first branch and bound (BFBnB) (Malone et al. 2011).

In A* (Hart, Nilsson, and Raphael 1968), an admissible heuristic function is used to calculate a lower bound on the cost from a node U in the order graph to *goal*. An f-cost is calculated for U by summing the cost from *start* to U (called $g(U)$) and the lower bound from U to *goal* (called $h(U)$). For BNSL, $g(U)$ corresponds to the score of the subnetwork over the variables U , and $h(U)$ estimates the score of the remaining variables. So $f(U) = g(U) + h(U)$. The f-cost provides an optimistic estimation on how good a path through U can be. The search maintains a list of nodes to be expanded sorted by f-costs called *open* and a list of already-expanded nodes called *closed*. Initially, *open* contains just *start*, and *closed* is empty. Nodes are then expanded from *open* in best-first order according to f-costs. Expanded nodes are added to *closed*. As better paths to nodes are discovered, they are added to *open*. Upon expanding *goal*, the shortest path from *start* to *goal* has been found.

In AWA* (Aine, Chakrabarti, and Kumar 2007), a sliding window search strategy is used to explore the order graph over a number of iterations. During each iteration, the algorithm uses a fixed window size, w , and tracks the layer l of the deepest node expanded. For the order graph, the layer of a node corresponds to the number of variables in its subnetwork. Nodes are expanded in best-first order as usual by A*; however, nodes selected for expansion in a layer less than $l - w$ are instead *frozen*. A path to *goal* is found in each iteration, which gives an upper bound solution. After finding the path to *goal*, the window size is increased by 1 and the frozen nodes become *open*. The iterative process continues until no nodes are frozen during an iteration, which means the upper bound solution is optimal. Alternatively, the search can be stopped early if a resource bound, such as running time, is exceeded; the best solution found so far is output.

In BFBnB (Zhou and Hansen 2006), nodes are expanded one layer at a time. Before beginning the BFBnB search, a quick search strategy, such as AWA* for a few iterations or greedy hill climbing, is used to find a “good” network and its score. The score is used as an upper bound. During the BFBnB search, any node with an f-cost greater than the upper bound can safely be pruned.

Yuan *et al.* (2011) gave a simple heuristic function. Later, tighter heuristics based on pattern databases were developed (Yuan and Malone 2012). All of the heuristics were shown to be *admissible*, i.e., to always give a lower bound on the cost from U to *goal*. Furthermore, the heuristics

have been shown to be *consistent*, which is a property similar to non-negativity required by Dijkstra’s algorithm. Consistent heuristics always underestimate the cost of the path between *any* two nodes (Edelkamp and SchrodL 2012). Primarily, in A*, consistency ensures that the first time a node is expanded, the shortest path to that node has been found, so no node ever needs to be re-expanded.

3 LEARNING UNDER POPS CONSTRAINTS

The main contribution of our current work focuses on taking advantage of the implicit information encoded in the POPS. We will first motivate our approach using a simple example and then describe the technical details.

3.1 A SIMPLE EXAMPLE

Table 1 shows the POPS for six variables. Based on these sets, we can see that not all variables can select all other variables as parents. For example, X_1 can only select X_2 as its parent (due to score pruning). We collect all of the potential parents for X_i by taking the union of all $PA \in \mathcal{P}_i$. Figure 2 shows the resulting *parent relation graph* for the POPS in Table 1. The parent relation graph includes an edge from X_j to X_i if X_j is a potential parent of X_i .

Naively, the complete order graph for six variables contains 2^6 nodes. However, from the parent relation graph, we see that none of $\{X_3, X_4, X_5, X_6\}$ can be a parent of X_1 or X_2 . Consequently, we can split the problem into two subproblems as shown in Figure 3: first, finding the shortest path from *start* to $\{X_1, X_2\}$, and then, finding the shortest path from $\{X_1, X_2\}$ to *goal*. Thus, the size of the search space is reduced to $2^2 + 2^4$.

3.2 ANCESTOR RELATIONS

This simple example shows that the parent relation graph can be used to prune the order graph without bounds. In general, we must consider *ancestor relations* to prune the order graph. In particular, if X_i can be an ancestor of X_j , and X_j cannot be an ancestor of X_i (due to local score pruning), then no node in the order graph which contains X_j but not X_i needs to be generated.

As a proof sketch, we can consider a node U which includes neither X_i nor X_j . If we add X_i and then X_j , then the cost from U to $U \cup \{X_i, X_j\}$ is $BestScore(X_i, U) + BestScore(X_j, U \cup \{X_i\})$. On the other hand, if we add X_j first, then the cost from U to $U \cup \{X_i, X_j\}$ is $BestScore(X_j, U) + BestScore(X_i, U \cup \{X_j\})$. However, due to the ancestor relations, we know that $BestScore(X_i, U \cup \{X_j\}) = BestScore(X_i, U)$. So, regardless of the order we add the two variables, X_i will have the same parent choices. If we add X_j first, though, then X_j

variable	POPS					
X_1	$\{X_2\}$	$\{\}$				
X_2	$\{X_1\}$	$\{\}$				
X_3	$\{X_1, X_2\}$	$\{X_2, X_6\}$	$\{X_1, X_6\}$	$\{X_2\}$	$\{X_6\}$	$\{\}$
X_4	$\{X_1, X_3\}$	$\{X_1\}$	$\{X_3\}$	$\{\}$		
X_5	$\{X_4\}$	$\{X_2\}$	$\{\}$			
X_6	$\{X_2, X_5\}$	$\{X_2\}$	$\{\}$			

Table 1: The POPS for six variables. The i th row shows \mathcal{P}_i .

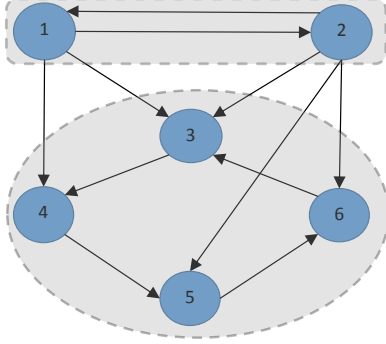


Figure 2: The parent relation graph constructed by aggregating the POPS in Table 1. The strongly connected components are surrounded by shaded shapes.

will have fewer choices. Therefore, adding X_j as a leaf first can never be better than adding X_i first (Yuan and Malone 2013).

3.3 POPS CONSTRAINTS PRUNING

We find the ancestor relations by constructing the parent relation graph and extracting its strongly connected components (SCCs). The SCCs of the parent relation graph form the *component graph*, which is a DAG (Cormen et al. 2001); each component graph node c_i corresponds to an SCC scc_i from the parent relation graph (which in turn corresponds to a set of variables in the Bayesian network). The component graph includes a directed edge from c_i to c_j if the parent relation graph includes an edge from a variable $X_i \in scc_i$ to $X_j \in scc_j$.

The component graph gives the ancestor constraints: if c_j is a descendent of c_i in the component graph, then variables in scc_j cannot be ancestors of variables in scc_i . Consequently, the component graph gives *POPS constraints* which allow the order graph to be pruned without considering bounds. In particular, the POPS constraints allow us to prune nodes in the order graph which do not respect the ancestor relations.

Tarjan’s algorithm (Tarjan 1972) extracts the SCCs from directed graphs, like the parent relation graph. We chose to use it because, in addition to its polynomial complexity, it

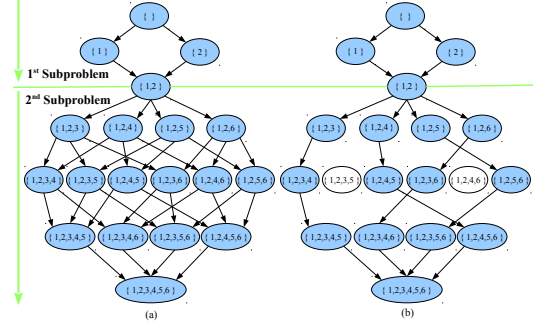


Figure 3: Order graphs after applying the POPS constraints. (a) The order graph after applying the POPS constraints once. (b) The order graph after recursively applying the POPS constraints on the second subproblem.

extracts the SCCs from the parent relation graph consistent with their topological order in the component graph. Consequently, all of the parent candidates of $X \in scc_i$ appear in $\mathbf{PC}_i = \cup_{k=1}^i scc_k$ ¹. After extracting the m SCCs, the search can be split into m independent subproblems: one for each SCC where $start_i$ is \mathbf{PC}_{i-1} and $goal_i$ is \mathbf{PC}_i . That is, during the i^{th} subproblem, we select the optimal parents for the variables in scc_i . Of course, $start_0 = \emptyset$ and $goal_m = \mathbf{V}$. The worst-case complexity of subproblem i is then $O(2^{|scc_i|})$. Figure 3(a) shows the pruned order graph resulting from the parent relation graph in Figure 2. In particular, it shows the first subproblem, from $\mathbf{PC}_0 = \emptyset$ to $\mathbf{PC}_1 = \{X_1, X_2\}$, and the second subproblem, from \mathbf{PC}_1 to $\mathbf{PC}_2 = \mathbf{V}$.

The (worst-case) size of the original order graph for n variables is as follows.

$$O(2^{|scc_1|} + \dots + |scc_m|) = O(2^n) \quad (3)$$

The worst-case size of the search space after splitting into subproblems using the SCCs is as follows.

$$O(2^{|scc_1|} + \dots + 2^{|scc_m|}) = O(m \cdot \max_{|i|} 2^{|scc_i|}) \quad (4)$$

That is, the complexity is at worst exponential in the size of the largest SCC. Consequently, our method can scale to

¹Depending on the structure of the component graph, this may be a superset of the parent candidates for X .

datasets with many variables if the largest SCC is of manageable size.

3.4 RECURSIVE POPS CONSTRAINTS PRUNING

As described in Section 3.3, the size of the search space for the i^{th} subproblem is $O(2^{|scc_i|})$, which can still be intractable for large SCCs. However, recursive application of the POPS constraints can further reduce the size of the search space. We refer to the constraints added by this strategy as *recursive POPS constraints*.

The intuition is the same as that behind the POPS constraints. As an example, consider the subproblem associated with scc_2 in Figure 3, which includes variables X_3 , X_4 , X_5 and X_6 . Naively, the order graph associated with this subproblem has $O(2^4)$ nodes. However, suppose we add variable X_3 as a leaf first. Then, the remaining variables split into three SCCs, and their order is completely determined. Similarly, selecting any of the other variables as the first to add as a leaf completely determines the order of the rest. Figure 3(b) shows the order graph after applying recursive POPS constraints.

In general, selecting the parents for one of the variables has the effect of removing that variable from the parent relation graph. After removing it, the remaining variables may split into smaller SCCs, and the resulting smaller subproblems can be solved recursively. These SCC checks can be implemented efficiently by again appealing to Tarjan’s algorithm. In particular, after adding variable X_i as a leaf from \mathbf{U} , we remove all of those variables from the parent relation graph. We then find the topologically first SCC and expand just the variables in that component. As we recursively explore the remaining variables, they will all eventually appear in the first SCC of the updated parent relation graph.

4 TOP- p POPS CONSTRAINT

As shown in Equation 4, the complexity of the search largely depends on the size of the largest strongly connected component. The recursive splitting described in Section 3.4 helps reduce this complexity, but for large, highly connected SCCs, the subproblems may still be too large to solve. For these cases, we can tradeoff between the complexity of the search and a bound on the optimality of the solution. In particular, rather than constructing the parent relation graph by aggregating *all* of the POPS, we can instead create the graph by considering only the best p POPS for each variable. We consider the minimization version of BNSL, so the best POPS are those with the lowest scores. This yields a set of parent candidates for each variable, and only POPS which are subsets of these parent candidates are retained. The empty set is always a subset of the parent candidates, so some DAG (e.g., the DAG with no edges) is always consistent with the resulting pruned set of POPS. We

call this score pruning strategy the *top- p POPS constraint*.

By removing some of the POPS in this manner, though, we can no longer guarantee to find the globally optimal Bayesian network. That is, this score pruning strategy is lossy. Despite losing the globally optimal guarantee, though, we can still offer a *bounded suboptimality* guarantee. In particular, suppose we apply the top- p POPS constraint and learn a BN N with score $s(N)$ in which X_i selects parents PA_i with score $s_i(PA_i)$. Additionally, suppose the best pruned parent set PA'_i for X_i has score $s_i(PA'_i)$. Then, the most improvement we could have in the score by including the pruned POPS for X_i is $\delta_i = \max(0, s_i(PA_i) - s_i(PA'_i))$. The max is necessary when the selected parent set is better than the best excluded parent set. Consequently, a suboptimality bound ϵ on the score of the unconstrained optimal network relative to $s(N)$ is as follows.

$$\epsilon = \frac{s(N)}{\sum_i (s_i(PA_i) - \delta_i)} \quad (5)$$

When ϵ is 1, N is the globally optimal network.

5 REDUCING THE SPACE REQUIREMENTS OF THE HEURISTIC

In this section we show that the POPS constraints can reduce the space requirements of the lower bound heuristic used during search.

A simple heuristic function was introduced for computing lower bounds for the order graph (Yuan and Malone 2013) which allows each remaining variable to choose optimal parents from all the other variables. This completely relaxes the acyclicity constraint on the BN structure. The heuristic was proven to be admissible, meaning it never overestimates the distance to *goal* (Yuan and Malone 2013). However, because of the complete relaxation of the acyclicity constraint, the simple heuristic may generate loose lower bounds.

5.1 THE k -CYCLE CONFLICT HEURISTIC

In (Yuan and Malone 2012), an improved heuristic function called *k-cycle conflict heuristic* was proposed which reduces the amount of relaxation. The idea is to divide the variables into multiple groups with a size up to k and enforce acyclicity within each group while still allowing cycles between the groups. Each group (subset of variables) is called a *pattern*. One approach to creating the patterns is to divide the variables \mathbf{V} into l approximately equal-sized static subsets \mathbf{V}_i (typically $l = 2$, so $k = n/2$). For each \mathbf{V}_i , a pattern database h_i is created by performing a breadth-first search in a “reverse” order graph in which *start* is \mathbf{V} and *goal* is \mathbf{V}_i . A node \mathbf{U}' in the graph is expanded by removing each of the variables $X \in \mathbf{V}_i$.

An arc from $U' \cup \{X\}$ to U' corresponds to selecting the best parents for X from among U' and has a cost of $BestScore(X, U')$. The optimal g cost for node U' gives the cost of the pattern $V \setminus U'$. The patterns from different groups are guaranteed to be mutually exclusive, so the heuristic value of a node U in the order graph is the sum of the pattern costs for the variables remaining in each partition. That is, $h(U) = \sum_i^l h_i(V_i \cap (V \setminus U))$. This approach is a *statically-partitioned additive pattern database heuristic* (Felner, Korf, and Hanan 2004) referred to as *static pattern databases*. Static pattern databases were shown to be consistent (Yuan and Malone 2013).

5.2 CREATING PATTERN DATABASES FOR SUBPROBLEMS

As described in Section 3, the search is split into an independent subproblem for each SCC. Furthermore, using Tarjan's algorithm, the SCCs are ordered according to their topological order in the component graph. Consequently, we construct static pattern databases using a similar strategy as before. Namely, each SCC is partitioned into l groups $scc_i = scc_{i1} \dots scc_{il}$ (typically $l = 2$). For each partition, a pattern database h_{ik} is created. For h_{ik} , the pattern costs are calculated using a breadth-first search in a reverse order graph in which *start* is $PC_{i-1} \cup scc_{ik}$ and *goal* is PC_{i-1} . The arc costs in this graph are $BestScore(X, (\bigcup_{j \neq k} scc_{ij}) \cup U)$. Thus, the heuristic value from U to PC_i , referred to as h_1 , is as follows.

$$h_1(U) = \sum_k^l h_{ik}(scc_{ik} \cap (V \setminus U)) \quad (6)$$

The pattern databases are constructed at the beginning of the search based on the parent relation graph. That is, new pattern databases are not created for recursive subproblems.

The pattern databases based on the SCCs are typically smaller than those previously proposed for the entire space. The space complexity of pattern databases created based on l balanced partitions is $O(l \cdot 2^{n/l})$. On the other hand, the space complexity of pattern databases created based on l balanced partitions separately for m SCCs of size $O(\max_{|scc_i|} 2^{|scc_i|})$ is $O(m \cdot \max_{|scc_i|} 2^{|scc_i|/l})$. Thus, the space complexity of the pattern databases based on the SCCs is less than that based on the balanced partitions alone, unless there is only one SCC. In that case, the space complexity is the same.

5.3 CALCULATING THE HEURISTIC VALUE

The heuristic value for node U in the subproblem for scc_i is calculated in two steps. We first calculate $h_1(U)$, the heuristic value from U to PC_i , using the the pattern databases described in Section 5.2. Second, we calculate

$h_2(U)$, the estimated distance from PC_i to V , as follows.

$$h_2(U) = \sum_{j=i+1}^m h_1(scc_j) \quad (7)$$

That is, the h_2 value is the sum of h_1 values for the start nodes of the remaining subproblems. Due to the POPS constraints, none of these variables will have been added as leaves when considering the i^{th} subproblem. The total heuristic value is then $h'(U) = h_1(U) + h_2(U)$. The h_2 values are the same for all nodes in the i^{th} subproblem, so they can be precomputed.

Theorem 1. *The new heuristic h' is consistent.*

Proof. We prove the theorem by showing that both h_1 and h_2 are consistent. The consistency of h_1 follows from the consistency of the static pattern databases (Yuan and Malone 2013). The h_2 value is a sum of h_1 values for mutually exclusive patterns, so it is also consistent. Therefore, the entire heuristic is consistent. \square

6 RELATED WORK

The parent relation graph is, in effect, a *directed superstructure*. Consequently, the work presented in this paper is quite related to the work dealing with superstructures. To the best of our knowledge, Ordyniak and Szeider (2013) are the only other authors to consider *directed* superstructures. They prove that BNSL is solvable in polynomial time for acyclic directed superstructures; our algorithm agrees with this theoretical result because, if the parent relation graph is acyclic, then it will have n SCCs of size 1. Thus, the complexity of our algorithm would be $O(n)$.

The work on undirected superstructures, e.g., (Perrier, Imoto, and Miyano 2008), is also related to our work. Any undirected superstructure can be translated into a parent relation graph by replacing the undirected edges in the superstructure with directed edges in both directions. However, edges directed in only one direction give an order to the SCCs which further reduce the search space. So, our algorithm leverages all of the information available from the undirected superstructure, but further makes use of constraints those structures cannot express.

Recently, Parviainen and Koivisto (2013) explored precedence constraints, which are similar to our POPS constraints. In their work, ideals of partial orders on the variables are used to reduce the search space of dynamic programming for BNSL. This approach is similar in spirit to our use of the component graph to reduce the search space. In fact, the component graph could be used to reduce the search space of dynamic programming. However, after selecting the ideals, they are fixed. So the recursive decomposition described in Section 3.4 is not compatible with the ideals formulation. Experimentally, we show that the

recursive application of constraints is important for some datasets.

Integer linear programming (ILP) (Bartlett and Cussens 2013) is another successful strategy for BNSL. A recent study (Malone et al. 2014) found that the performances of ILP and heuristic search are largely orthogonal, particularly with respect to the number of POPS. Consequently, this work has focused on improvements to heuristic search. Nevertheless, the component graph is readily applicable to ILP by similarly creating subproblems and solving them with independent ILP instances. Indeed, an interesting avenue for future research is to dynamically select between ILP and heuristic search for each subproblem.

6.1 EXPERT KNOWLEDGE CONSTRAINTS

The formulation of BNSL as an optimization over POPS gives a natural method for including expert knowledge in the form of hard constraints on the structure to be learned, such as those proposed by, e.g., (de Campos and Ji 2011). In particular, given expert knowledge about required or forbidden parent relationships and maximum parent set cardinalities, we omit POPS which violate these constraints. The POPS constraints automatically prune the parts of the search space which violate the expert knowledge.

In general, hard expert knowledge constraints are lossy because they could disallow parent sets which would appear in an optimal structure based solely on the data and scoring function. Nevertheless, we still consider the network learned under expert knowledge constraints as optimal. For cases in which we use expert knowledge constraints and the top- p POPS constraint, parent sets disallowed by the expert knowledge constraints are not considered in the suboptimality bound calculation in Equation 5.

7 EMPIRICAL EVALUATION

In order to evaluate the efficacy of the POPS constraints and top- p POPS constraint, we ran a set of experiments on benchmark datasets from the UCI machine learning repository² and the Bayesian network repository³. We generated 1,000 records from the benchmark networks in the repository using logic sampling. The experiments were performed on an IBM System x3850 X5 with 16 2.67GHz Intel Xeon processors and 512G RAM; 1TB disk space was used. Our code is available online⁴.

Several heuristic search algorithms have been adapted for BNSL. We chose to evaluate A* (Yuan, Malone, and Wu 2011) because of its guarantee to expand a minimal number of nodes; AWA* (Malone and Yuan 2013) because it

has been shown to find high quality, often optimal, solutions very quickly; and breadth-first branch and bound (BF-BnB) (Malone et al. 2011) because it has been shown to scale to larger datasets by using external memory. We used MDL as the scoring function. In all cases, we used static pattern databases; the variable groups were determined by partitioning the parent relation graph after applying the top- $p = 1$ POPS constraint (Fan, Yuan, and Malone 2014). Pattern database construction occurs only once after constructing the parent relation graph.

7.1 POPS CONSTRAINTS

We first tested the effect of the POPS constraints, which always guarantee learning the globally optimal structure. Table 2 compares the original version of each algorithm to versions using the POPS constraints.

We first considered three variants of A*: a basic version not using POPS constraints; a version using the POPS constraints but not applying them recursively as described in Section 3.4; and a version which uses the recursive POPS constraints. As the table shows, the versions of A* augmented with the POPS constraints always outperform the basic version. The improvement in running time ranges from two times on several of the datasets to over an order of magnitude on three of the datasets. Additionally, the basic version is unable to solve Mildew, Soybean and Barley within the time limit (30 minutes); however, with the POPS constraints, all of the datasets are easily solved within the limit. The number of nodes expanded, and, hence, memory requirements, are similarly reduced.

The recursive POPS constraints always reduce the number of nodes expanded⁵. However, it sometimes increases the running time. The overhead of Tarjan’s algorithm to recursively look for SCCs is small; however, in some cases, such as when the parent relation graph is dense, the additional work yields minimal savings. In these cases, despite the reduction in nodes expanded, the running time may increase.

On the other hand, when the parent relation graph is sparse, the advantages of the recursive POPS constraints are sometimes more pronounced. For example, the running time of Mildew is reduced in half by recursively applying POPS constraints. Most networks constructed by domain experts, including those evaluated in this study, are sparse. Our analysis shows that these datasets also yield sparse parent relation graphs. Thus, our results suggest that the recursive constraints are sometimes effective when the generative process of the data is sparse. The overhead of looking for the recursive POPS constraints is minimal, and it sometimes offers substantial improvement for sparse generative processes. So we always use it in the remaining experiments.

⁵For some datasets, the precision shown in the table is too coarse to capture the change.

²<http://archive.ics.uci.edu/ml>

³<http://compbio.cs.huji.ac.il/Repository/>

⁴<http://url.cs.qc.cuny.edu/software/URLearning.html>

Name	Dataset					Results							
	n	N	POPS	Density	PD (s)	A*	A*, O	A*,R	AWA*	AWA*,R	BFBnB	BFBnB,R	
Mushroom	23	8124	13025	0.87	0.15	Time (s) Nodes	0.74 0.05	0.41 0.04	0.67 0.04	0.75 0.06	0.47 0.05	0.61 0.06	0.78 0.04
Autos	26	159	2391	0.75	0.17	Time (s) Nodes	46.62 3.26	20.93 1.63	26.76 1.63	44.70 4.92	19.86 2.47	11.24 3.26	6.68 1.63
Insurance*	27	1000	560	0.35	0.21	Time (s) Nodes	98.08 7.83	52.81 3.92	50.97 3.77	118.23 14.51	66.75 6.51	47.46 8.16	28.58 3.77
Water*	32	1000	4022	0.24	0.49	Time (s) Nodes	14.10 1.59	0.03 0.02	0.03 0.02	14.10 1.59	0.03 0.01	32.82 7.10	0.80 0.01
Mildew*	35	1000	360	0.16	0.50	Time (s) Nodes	OT OT	5.20 0.56	2.33 0.37	OT OT	3.71 0.44	OT OT	3.18 0.36
Soybean	36	307	5926	0.58	0.54	Time (s) Nodes	OT OT	435.65 9.78	511.55 9.64	OT OT	526.13 11.36	OT OT	1230.41 129.77
Alarm*	37	1000	672	0.16	1.39	Time (s) Nodes	76.51 2.75	6.47 0.33	4.06 0.24	46.98 3.49	4.80 0.30	22.32 2.75	3.83 0.24
Bands	39	277	892	0.26	2.03	Time (s) Nodes	109.75 3.63	0.39 0.03	0.47 0.03	74.02 3.99	0.40 0.03	249.04 41.81	1.34 0.03
Spectf	45	267	610	0.24	43.46	Time (s) Nodes	89.08 2.26	90.62 2.26	92.17 2.17	44.41 3.17	36.79 3.17	32.34 2.53	29.59 2.44
Barley*	48	1000	634	0.1	0.73	Time (s) Nodes	OT OT	2.51 0.64	1.28 0.08	OT OT	1.10 0.21	OT OT	1.85 0.08

Table 2: The number of expanded nodes (in millions) and running time (in seconds) of A*, AWA* and BFBnB with/without the POPS constraints on a set of benchmark datasets. “n” gives the number of variables in the dataset, “N” gives the number of records in the dataset, “POPS” gives the number of POPS after lossless pruning, “Density” gives the density of the parent relation graph constructed from the POPS (defined as the number of edges divided by the number of possible edges), and “PD” gives the time (in seconds) to construct the pattern database. “A*,O” gives the statistics for A* using the POPS constraints, but not applying them recursively. “A*,R” gives the statistics for A* using the recursive POPS constraints. Similarly, “AWA*”, “AWA*, R”, “BFBnB” and “BFBnB, R” refer to the respective basic algorithms or the algorithm using recursive POPS constraints. “*” indicates the dataset was generated from a repository network using logic sampling; all other datasets are from UCI. OT means out of time (30 minutes).

The anytime window A* algorithm enjoyed improvements similar to those seen in A*. As the table shows, A* always expanded fewer nodes than AWA*; nevertheless, the run-times of AWA* are often shorter than those of A*. This is because AWA* performs a series of iterations, and *open* is cleared after each of those iterations. Consequently, the associated priority queue operations are often faster for AWA* than A*.

A key factor in the performance for BFBnB is the upper bound it uses for pruning. Previous results (Malone and Yuan 2013) have shown that AWA* is effective at finding high quality solutions quickly, so we found the bound by running AWA* for 5 seconds on datasets with less than 35 variables and 10 seconds for larger datasets. AWA* used the POPS constraints when BFBnB used them. BFBnB exhibited improvements in line with those for A* and AWA*.

7.2 TOP- p POPS CONSTRAINT

We tested AWA* on the dataset Hailfinder, which has 56 variables. Even when using the recursive POPS constraints, though, AWA* was unable to prove optimality within the 30-minute time limit. Therefore, we used this dataset to test the effect of the top- p POPS constraint by varying p from 1 to 13. The upper bound on p was set to 13 because AWA* was unable to complete within the time limit for $p = 13$.

Primarily, we evaluated the running time and associated suboptimality bound as we increased p (which has the effect of pruning fewer POPS). As Figure 4 (top) shows, the recursive order constraints are quite effective under the top- p POPS constraint; the constrained problems are solved in under 15 seconds for p up to 12, and the provable suboptimality bound calculated using Equation 5 decreases very

rapidly. This provable suboptimality between the learned network and global optimum is less than 1% even when p is only 7.

The suboptimality bound usually decreases as p increases. From $p = 7$ to $p = 8$, though, it slightly increases; the scores of the learned networks were the same (not shown). This is a result of equivalence classes of Bayesian networks. The suboptimality bound calculation in Equation 5 focuses on parent sets of individual variables, so it is sensitive to which member of an equivalence class the algorithm learns. Future work could investigate tightening the bound by considering all members in the same equivalence class as the learned network.

As mentioned, AWA* was unable to find the provably optimal network under the $p = 13$ constraint. Equation 4 suggests that the size of the largest SCC in the parent relation graph is a key factor in determining the difficulty of an instance of BNSL. However, the recursive POPS constraints offer the potential to split large SCCs after considering a few of their variables. Figure 4 (middle) shows that the size of the largest SCC does substantially increase from $p = 12$ to $p = 13$, which empirically confirms our theoretical result. Somewhat unexpectedly, though, the figure also shows that the density of the parent relation graph does not significantly increase as more POPS are included. So, at least in this case, despite the sparsity of the parent relation graph, the recursive order constraints are unable to break the large SCC into manageable subproblems. This result agrees with those in Table 2 which show that sparsity does not necessarily indicate the efficacy of the recursive POPS constraints.

In addition to the characteristics of the parent relation graph, we also considered the number of POPS included as p in-

creases. Figure 4 (bottom) shows that the number of included POPS follows a similar trend to the density of the parent relation graph. That is, even as p increases, more POPS are not necessarily included for all variables. This is because we already include all subsets of parent candidates from the top p POPS at earlier iterations. Additionally, the number of POPS (479 when $p = 13$) is quite small for this dataset, although the number of variables is relatively large (56). Previous studies (Malone and Yuan 2013) have shown that basic heuristic search methods struggle with datasets like this; however, when augmented with the POPS constraints, heuristic search very quickly finds a network that is provably quite close to optimal. This result clearly shows that the POPS constraints significantly expand the applicability of heuristic search-based structure learning.

Despite the inability of AWA* to find the provably optimal network under the top- p POPS constraint when $p = 13$, we can nevertheless take advantage of its anytime behavior to calculate a suboptimality bound. At each iteration, AWA* produces an optimal network with respect to its current window size. We can then use Equation 5 to bound the suboptimality of the learned network. In principle, this even suggests that we may be able to prove global optimality before completing the AWA* search, although this likely would require a tighter bound under the top- p POPS constraint than the naive one proposed in this paper.

Even for very small values of p , though the top- p POPS constraint results in networks provably very close to the globally optimal solution. In order to more thoroughly understand why such constrained problems still give provably very high quality solutions, we plotted the scores of the top p POPS for variable X_{37} from Hailfinder in Figure 5. The figure shows that the first 4 scores are much better than the remaining ones; consequently, the globally optimal network is more likely to include one of these parent sets for X_{37} than the others, which are much worse. Most of the other variables behaved similarly; consequently, p does not need to be very large to still encompass most of the parent selections in the globally optimal network.

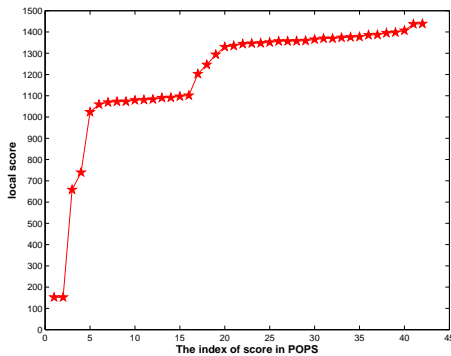


Figure 5: The POPS of variable X_{37} from Hailfinder, sorted in ascending order

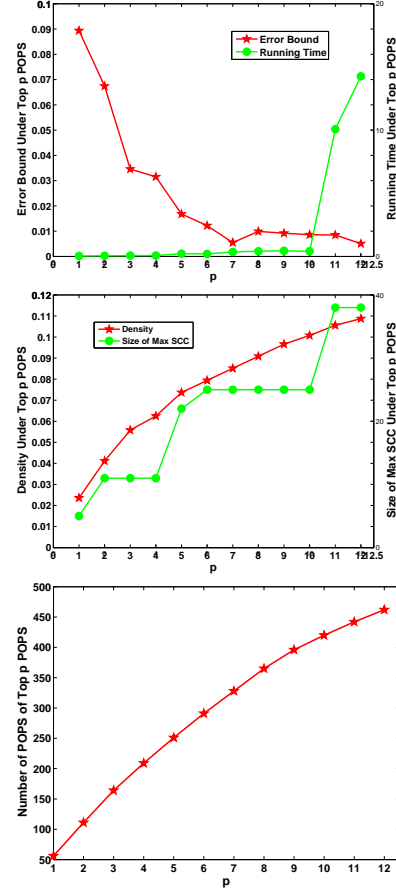


Figure 4: The behavior of Hailfinder under the top- p POPS constraint as p varies. (top) Running time and suboptimality (middle) Size of the largest SCC and density of the parent relation graph (bottom) Number of POPS included

8 CONCLUSION

In this work, we have shown how POPS constraints, which are implicit in the input to a BNSL instance, can significantly improve the performance of heuristic search on the problem. Other algorithms, such as integer linear programming, can also benefit from the POPS constraints. We also introduced the top- p POPS constraint and showed how it can be used to further take advantage of the POPS constraints while still providing guaranteed error bounds. Empirically, we showed that the POPS constraints are practically effective and that the top- p POPS constraint can yield provably very high quality solutions very quickly. Future work includes more thorough empirical evaluation and comparison with other BNSL techniques as well as investigation into conditions when the POPS constraints are most effective.

Acknowledgements This research was supported by NSF grants IIS-0953723, IIS-1219114 and the Academy of Finland (COIN, 251170).

References

- Aine, S.; Chakrabarti, P. P.; and Kumar, R. 2007. AWA*-a window constrained anytime heuristic search algorithm. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 2250–2255.
- Bartlett, M., and Cussens, J. 2013. Advances in Bayesian network learning using integer programming. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence*.
- Buntine, W. 1991. Theory refinement on Bayesian networks. In *Proceedings of the 7th Conference on Uncertainty in Artificial Intelligence*, 52–60.
- Chickering, D. M. 1996. Learning Bayesian networks is NP-complete. In *Learning from Data: Artificial Intelligence and Statistics V*, 121–130. Springer-Verlag.
- Cormen, T. H.; Stein, C.; Rivest, R. L.; and Leiserson, C. E. 2001. *Introduction to Algorithms*. McGraw-Hill Higher Education.
- Cussens, J. 2011. Bayesian network learning with cutting planes. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence*, 153–160.
- de Campos, C. P., and Ji, Q. 2011. Efficient learning of Bayesian networks using constraints. *Journal of Machine Learning Research* 12:663–689.
- Edelkamp, S., and Schrod, S. 2012. *Heuristic Search: Theory and Applications*. Morgan Kaufmann.
- Fan, X.; Yuan, C.; and Malone, B. 2014. Tightening bounds for Bayesian network structure learning. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*.
- Felner, A.; Korf, R.; and Hanan, S. 2004. Additive pattern database heuristics. *Journal of Artificial Intelligence Research* 22:279–318.
- Hart, P. E.; Nilsson, N. J.; and Raphael, B. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions On Systems Science And Cybernetics* 4(2):100–107.
- Heckerman, D.; Geiger, D.; and Chickering, D. M. 1995. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning* 20:197–243.
- Heckerman, D. 1998. A tutorial on learning with Bayesian networks. In Jordan, M., ed., *Learning in Graphical Models*, volume 89 of *NATO ASI Series*. Springer Netherlands. 301–354.
- Jaakkola, T.; Sontag, D.; Globerson, A.; and Meila, M. 2010. Learning Bayesian network structure using LP relaxations. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*.
- Koivisto, M., and Sood, K. 2004. Exact Bayesian structure discovery in Bayesian networks. *Journal of Machine Learning Research* 5:549–573.
- Lam, W., and Bacchus, F. 1994. Learning Bayesian belief networks: An approach based on the MDL principle. *Computational Intelligence* 10:269–293.
- Malone, B., and Yuan, C. 2013. Evaluating anytime algorithms for learning optimal Bayesian networks. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence*.
- Malone, B.; Yuan, C.; Hansen, E.; and Bridges, S. 2011. Improving the scalability of optimal Bayesian network learning with external-memory frontier breadth-first branch and bound search. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence*, 479–488.
- Malone, B.; Kangas, K.; Järvisalo, M.; Koivisto, M.; and Myllymäki, P. 2014. Predicting the hardness of learning Bayesian networks. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*.
- Ordyniak, S., and Szeider, S. 2013. Parameterized complexity results for exact Bayesian network structure learning. *Journal of Artificial Intelligence Research* 46:263–302.
- Ott, S.; Imoto, S.; and Miyano, S. 2004. Finding optimal models for small gene networks. In *Pacific Symposium on Biocomputing*, 557–567.
- Parviainen, P., and Koivisto, M. 2013. Finding optimal Bayesian networks using precedence constraints. *Journal of Machine Learning Research* 14:1387–1415.
- Perrier, E.; Imoto, S.; and Miyano, S. 2008. Finding optimal Bayesian network given a super-structure. *Journal of Machine Learning Research* 9:2251–2286.
- Silander, T., and Myllymäki, P. 2006. A simple approach for finding the globally optimal Bayesian network structure. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*.
- Singh, A., and Moore, A. 2005. Finding optimal Bayesian networks by dynamic programming. Technical report, Carnegie Mellon University.
- Tarjan, R. 1972. Depth-first search and linear graph algorithms. *SIAM Journal on Computing* 1(2):146–160.
- Yuan, C., and Malone, B. 2012. An improved admissible heuristic for finding optimal Bayesian networks. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence*.
- Yuan, C., and Malone, B. 2013. Learning optimal Bayesian networks: A shortest path perspective. *Journal of Artificial Intelligence Research* 48:23–65.
- Yuan, C.; Malone, B.; and Wu, X. 2011. Learning optimal Bayesian networks using A* search. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*.
- Zhou, R., and Hansen, E. A. 2006. Breadth-first heuristic search. *Artificial Intelligence* 170:385–408.

Bisimulation Metrics are Optimal Value Functions

Norm Ferns *

Département d'Informatique
École Normale Supérieure
45 rue d'Ulm, F-75230 Paris Cedex 05, France

Doina Precup †

School of Computer Science
McGill University
Montréal, Canada, H3A 0E7

Abstract

Bisimulation is a notion of behavioural equivalence on the states of a transition system. Its definition has been extended to Markov decision processes, where it can be used to aggregate states. A bisimulation metric is a quantitative analog of bisimulation that measures how similar states are from the perspective of long-term behavior. Bisimulation metrics have been used to establish approximation bounds for state aggregation and other forms of value function approximation. In this paper, we prove that a bisimulation metric defined on the state space of a Markov decision process is the optimal value function of an optimal coupling of two copies of the original model. We prove the result in the general case of continuous state spaces. This result has important implications in understanding the complexity of computing such metrics, and opens up the possibility of more efficient computational methods.

1 INTRODUCTION

Markov decision processes (MDPs) are a popular mathematical model for sequential decision-making under uncertainty (Puterman, 1994; Sutton & Barto, 2012). Many standard solution methods are based on computing or learning the optimal value function, which reflects the expected return one can achieve in each state by choosing actions according to the optimal policy. In finite MDPs, the optimal value function is guaranteed to be unique, and has at least one deterministic optimal policy associated with it.

A major challenge is how to deal with large, possibly continuous, state spaces, known more colourfully as *the curse of dimensionality* or *the state-space explosion problem*.

* Norm Ferns' contribution was partially supported by the AbstractCell ANR-Chair of Excellence.

† Doina Precup's contribution was supported by NSERC.

Briefly, the number of parameters necessary to represent the value function scales exponentially with the number of state variables. In response to this issue, a number of researchers have advocated the use of metrics, which can be used to determine similarity between states, and cluster them accordingly. Ideally, one would like such a clustering to reflect similarity among states in terms of the value function, which reflects the long-term cumulative reward.

In the formal verification community, a similar problem arises in the analysis of transition systems, in which one wants to establish long-term properties (e.g., the probability that the system may enter a faulty state, or that a certain trajectory would terminate). Many researchers advocate tackling such problems by using approximation metrics based on strong probabilistic bisimulation. Bisimulation is a conservative behavioural equivalence between states: states that are bisimilar will have the same long-term behaviour (Larsen & Skou, 1991; Givan et al., 2003). Corresponding metrics are useful in order to measure state similarity, and are used both to directly aggregate system states and more generally to assess the quality of an approximation. Abate (2012) surveys historical and more recent developments in this area.

In the context of MDPs, such metrics - henceforth known as *bisimulation metrics* - were developed in (Ferns et al., 2004; Ferns et al., 2005; Ferns et al., 2011) based on the work of Desharnais et al. (2002; 2001a) for a related Markov transition system. In (Ferns et al., 2006), the authors experimentally compared several methods for estimating these metrics on small finite MDPs, with a Monte Carlo approach outperforming the others. However, the analyses therein were limited, lacking, for example, any sample complexity results.

The purpose of this work is to strengthen and unify theoretical and practical results for bisimulation metrics on a given MDP by showing that they are in fact the optimal value functions of an optimal coupling of that MDP with itself (Theorem 3.3). We establish this result in the general setting of continuous-state MDPs. To our knowledge, this is an original result, which both improves our under-

standing of bisimulation metrics in general, and opens up avenues of attack for more efficient computation.

The paper is organized as follows. In Section 2, we provide a brief summary of MDPs, optimal control theory, and bisimulation metrics on continuous spaces. In Section 3, we use a measurable selection theorem to prove the main result, and in Section 4 we relate the results we present to existing work within the artificial intelligence and formal verification communities. Finally, in Section 5, we discuss the implications of our result and directions for future research.

2 BACKGROUND

Since we deal primarily with uncountably infinite state spaces, we must take into account the tension between imposing the right amount of structure on a space for general theoretical usefulness and imposing the right amount of structure for useful practical applications. For that reason, much of the work on Markov processes has been cast in the setting of Polish spaces. The introductory chapter of (Doberkat, 2007) contains a self-contained exposition of probabilities on Polish spaces in the context of computer science. A more comprehensive mathematical description can be found in (Srivastava, 2008). By contrast, a gentler introduction to probabilities on continuous spaces can be found in the first four chapters of (Folland, 1999). We refer the reader to these three sources for the basic mathematical definitions that we present throughout.

2.1 PROBABILITIES ON METRIC SPACES

A *Polish metric space* is a complete, separable metric space. A *Polish space* is a topological space that is homeomorphic to a Polish metric space. A *standard Borel space* is a measurable space that is Borel isomorphic to a Polish space.

If (X, τ) is a topological space, then $\mathbb{C}^b(X)$ is the set of continuous bounded real-valued functions on X . If (X, \mathcal{B}_X) is a standard Borel space then we denote by $\mathbb{B}^b(X)$ the space of bounded measurable real-valued functions on X , and by $\mathbb{P}(X)$ the set of probability measures on X ; note that the latter is also a standard Borel space (Giry, 1982).

2.2 DISCOUNTED MARKOV DECISION PROBLEMS

Let (X, \mathcal{B}_X) and (Y, \mathcal{B}_Y) be standard Borel spaces. A *Markov kernel* is a Borel measurable map from (X, \mathcal{B}_X) to $(\mathbb{P}(Y), \mathcal{B}_{\mathbb{P}(Y)})$. Equivalently, K is a Markov kernel from X to Y iff $K(x)$ is a probability measure on (Y, \mathcal{B}_Y) for each $x \in X$, and $x \mapsto K(x)(B)$ is a measurable map for each $B \in \mathcal{B}_Y$.

Remark 1. *The use of the term kernel here should not be confused with the usual meaning, that being the set of points in the domain of a real-valued function that send the function to 0. We will refer to kernel in both senses throughout, with the meaning clear from the context.*

We will denote the set of all Markov kernels from X to Y by $\llbracket X \rightarrow \mathbb{P}(Y) \rrbracket$ and simply write “ K is a Markov kernel on X ” when it is implicitly assumed that $Y = X$. If I is an index set and $K = (K_i)_{i \in I}$ is an I -indexed collection of Markov kernels on X , we will say that “ K is a labelled Markov kernel on X ”. Such kernels play the role of transition relations in stochastic transition systems with continuous state spaces.

A *Markov decision process (MDP)* is a tuple $(S, \mathcal{B}_S, A, (P_a)_{a \in A}, r)$, where (S, \mathcal{B}_S) is a standard Borel space, A is a finite set of actions, $r : A \times S \rightarrow \mathbb{R}$ is a bounded measurable reward function, and for $a \in A$, P_a is a Markov kernel on S .

For each $a \in A$, we denote by $r_a : S \rightarrow \mathbb{R}$ the function defined by $r_a(s) = r(a, s)$, and for each $a \in A$ and $s \in S$. We use functional notation for integration with respect to $P_a(s)$, i.e. the integral of $f \in \mathbb{B}^b(S)$ with respect to $P_a(s)$ will be written as $P_a(s)(f)$.

A Markov decision process along with an optimality criterion is known as a *Markov decision problem*. In this work, we focus on Markov decision problems with the *expected total discounted reward optimality criterion*, which we now briefly describe based on (Hernández-Lerma & Lasserre, 1996) and especially Section 8.3 of (Hernández-Lerma & Lasserre, 1999). We rely on these sources instead of others who may be more familiar to the AI audience because they treat the infinitely uncountable state space setting. We direct the reader to these sources for full details.

Fix an MDP $\mathcal{M} = (S, \mathcal{B}_S, A, (P_a)_{a \in A}, r)$ and a discount factor $\gamma \in (0, 1)$. Let $t \in \mathbb{N}$. Then H_t , the family of *histories up to time t* , is defined by $H_0 = S$ and $H_{t+1} = H_t \times (A \times S)$ for $t \in \mathbb{N}$. An element $h_t = (s_0, a_0, \dots, s_{t-1}, a_{t-1}, s_t) \in H_t$ is called a *t -history*. A *randomized control policy* is a sequence of Markov kernels $\pi = (\pi_t)_{t \in \mathbb{N}}$ such that $\pi_t \in \llbracket H_t \rightarrow \mathbb{P}(A) \rrbracket$ for all $t \in \mathbb{N}$. The set of all policies is denoted by Π . A policy $\pi = (\pi_t)_{t \in \mathbb{N}}$ is said to be a *randomized stationary policy* if there exists a Markov kernel $\varphi \in \llbracket S \rightarrow \mathbb{P}(A) \rrbracket$ such that $\pi_t(h_t) = \varphi(s_t)$ for all $h_t \in H_t, t \in \mathbb{N}$, and a *deterministic stationary policy* if there exists a measurable selector f for $S \times A$ such that $\pi_t(h_t)$ is the Dirac measure at the point $f(s_t) \in A$ for all $h_t \in H_t, t \in \mathbb{N}$. We denote the sets of randomized stationary policies and deterministic stationary policies by Π_{RS} and Π_{DS} respectively and note that $\Pi_{DS} \subseteq \Pi_{RS} \subseteq \Pi$.

Let $\pi \in \Pi$ be a policy on \mathcal{M} . The γ -discounted value function for π , $V_\gamma(\pi)$, is defined by $V_\gamma(\pi)(s) =$

$\mathbb{E}_s^\pi[\sum_{t=0}^\infty \gamma^t r(a_t, x_t)]$ for all $s \in S$, where \mathbb{E}_s^π is the expectation taken with respect to the system dynamics when starting in state s and following policy π . The goal of this Markov decision problem is to find a policy whose value function dominates all others. Toward that end, one defines the γ -discounted optimal value function, V_γ^* , by $V_\gamma^*(s) = \sup_{\pi \in \Pi} V_\gamma(\pi)(s)$ for all $s \in S$, and the Bellman operator with respect to γ , $T_\gamma : \mathbb{B}^b(S) \rightarrow \mathbb{B}^b(S)$, by $T_\gamma(v)(s) = \max_{a \in A} [r_a(s) + \gamma \cdot P_a(s)(v)]$ for all $s \in S$. The following can be found within Theorem 8.3.6 and its preceding remarks in (Hernández-Lerma & Lasserre, 1999).

Theorem 2.1 (Value Iteration). *Define $(v_n)_{n \in \mathbb{N}} \subseteq \mathbb{B}^b(S)$ by $v_0(s) = 0$ for all $s \in S$ and $v_{n+1} = T_\gamma(v_n)$ for all $n \in \mathbb{N}$. Then the optimal value function V_γ^* is the unique solution in $\mathbb{B}^b(S)$ to the Bellman optimality equation $v = T_\gamma(v)$; $(v_n)_{n \in \mathbb{N}}$ converges uniformly to V_γ^* with $\|v_n - V_\gamma^*\| \leq \gamma^{-n}(1 - \gamma)$ for all $n \in \mathbb{N}$ and where $\|\cdot\|$ is the uniform norm; and there exists a deterministic stationary optimal policy $\pi^* \in \Pi_{DS}$ such that $V_\gamma^* = V_\gamma(\pi^*)$.*

We note that although Theorem 2.1 implies it is sufficient to search Π_{DS} for an optimal policy, in practice it is often useful to work with the larger class Π_{RS} . On the other hand, for more general theoretical considerations, e.g. other optimality criteria, we may need to consider all of Π .

2.3 BISIMULATION

We present bisimulation for MDPs as outlined in (Ferns et al., 2011).

Given an equivalence relation R on a measurable space (S, Σ) , a subset X of S is said to be R -closed if X is a union of R -equivalence classes. We write $\Sigma(R)$ for the set of those Σ -measurable sets that are also R -closed.

Let $(S, \mathcal{B}_S, A, (P_a)_{a \in A}, r)$ be an MDP. An equivalence relation R on S is a *bisimulation relation* if it satisfies $sRs' \iff$ for every $a \in A$, $r_a(s) = r_a(s')$ and for every $X \in \Sigma(R)$, $P_a(s)(X) = P_a(s')(X)$. *Bisimilarity* is the largest of the bisimulation relations.

2.4 THE KANTOROVICH METRIC

In order to define bisimulation metrics for MDPs, we first need to recall the definition and properties of the Kantorovich metric between distributions, which can be found in (Villani, 2003).

Definition 1 (Kantorovich Metric). *Let S be a Polish space, h a bounded pseudometric on S that is lower semi-continuous on $S \times S$ with respect to the product topology, and let $\text{Lip}(h)$ be the set of all $f \in \mathbb{B}^b(S)$ that satisfy the Lipschitz condition $f(x) - f(y) \leq h(x, y)$ for every $x, y \in S$. Let $P, Q \in \mathbb{P}(S)$. Then the Kantorovich metric $\mathcal{K}(h)$ is defined by $\mathcal{K}(h)(P, Q) = \sup_{f \in \text{Lip}(h)} (P(f) - Q(f))$.*

The Kantorovich metric is an infinite linear program and has a dual described in terms of couplings of probability measures.

Definition 2 (Coupling). *Let (X, \mathcal{B}_X) and (Y, \mathcal{B}_Y) be standard Borel spaces, and let $(X \times Y, \mathcal{B}_X \otimes \mathcal{B}_Y)$ be the product space. Let $\mu \in \mathbb{P}(X)$, $\nu \in \mathbb{P}(Y)$, and $\lambda \in \mathbb{P}(X \times Y)$. Then λ is a coupling of μ and ν if and only if its marginals on X and Y are μ and ν , respectively. We denote the set of all couplings of μ and ν by $\Lambda(\mu, \nu)$, i.e., $\lambda \in \Lambda(\mu, \nu) \iff \lambda(E \times Y) = \mu(E)$ and $\lambda(X \times F) = \nu(F)$ for all $E \in \mathcal{B}_X, F \in \mathcal{B}_Y$.*

The following is found within Section 1.1.1 of (Villani, 2003).

Lemma 2.2. *Let X and Y be Polish spaces and let μ and ν belong to $\mathbb{P}(X)$ and $\mathbb{P}(Y)$, respectively. Then $\lambda \in \Lambda(\mu, \nu)$ if and only if for every $(\varphi, \psi) \in \mathbb{C}^b(X) \times \mathbb{C}^b(Y)$*

$$\begin{aligned} \int_{X \times Y} [\varphi(x) + \psi(y)] \lambda(dx, dy) \\ = \int_X \varphi(x) \mu(dx) + \int_Y \psi(y) \nu(dy). \end{aligned}$$

In Section 3, we'll make use of the following simple lemma.

Lemma 2.3. *Let X and Y be Polish spaces and let μ and ν belong to $\mathbb{P}(X)$ and $\mathbb{P}(Y)$, respectively. Then $\Lambda(\mu, \nu)$ is a closed subset of $\mathbb{P}(X \times Y)$.*

Proof. Let $(\lambda_n)_{n \in \mathbb{N}} \subseteq \Lambda(\mu, \nu)$ be a sequence converging to some $\lambda \in \mathbb{P}(X \times Y)$ in the weak topology. Let $(\varphi, \psi) \in \mathbb{C}^b(X) \times \mathbb{C}^b(Y)$. Then

$$\begin{aligned} \int_{X \times Y} [\varphi(x) + \psi(y)] \lambda(dx, dy) \\ = \lim_{n \rightarrow \infty} \left(\int_{X \times Y} [\varphi(x) + \psi(y)] \lambda_n(dx, dy) \right) \\ = \lim_{n \rightarrow \infty} \left(\int_X \varphi(x) \mu(dx) + \int_Y \psi(y) \nu(dy) \right) \\ = \int_X \varphi(x) \mu(dx) + \int_Y \psi(y) \nu(dy). \end{aligned}$$

Here we have used the definition of weak convergence, as well as Lemma 2.2 for each λ_n . It follows from the same lemma that $\lambda \in \Lambda(\mu, \nu)$. \square

The following can be found in Theorem 1.3 and the proof of Theorem 1.14 in (Villani, 2003).

Theorem 2.4 (Kantorovich-Rubinstein Duality Theorem). *Assume the conditions of Definition 1. Then $\mathcal{K}(h)(P, Q)$ is equal to*

$$\sup_{f \in \text{Lip}(h, \mathbb{C}^b(S))} (P(f) - Q(f)) = \inf_{\lambda \in \Lambda(P, Q)} \lambda(h)$$

where $\text{Lip}(h, \mathbb{C}^b(S))$ denotes functions on S that are continuous and bounded, 1-Lipschitz with respect to h , and have range $[0, \|h\|]$. Moreover, the supremum and infimum are attained.

2.5 BISIMULATION METRICS

The following can be found in Theorem 3.12 of (Ferns et al., 2011) and Corollary 3 of (Ferns et al., 2014).

Theorem 2.5. *Let $\mathcal{M} = (S, \mathcal{B}_S, A, (P_a)_{a \in A}, r)$ be an MDP and let $c \in (0, 1)$ be a discount factor. Assume that the image of r is contained in $[0, 1]$. Then there exists a Polish topology τ generating \mathcal{B}_S such that for all $a \in A$, r_a is continuous with respect to τ and P_a is weakly continuous with respect to τ . Define $\theta^c : A \times (S \times S) \rightarrow [0, 1]$ by $\theta_a^c(x, y) = (1 - c)|r_a(x) - r_a(y)|$. Furthermore, let $\text{ls}\mathfrak{c}_m$ be the set of bounded pseudometrics on S that are lower semicontinuous on $S \times S$ endowed with the product topology induced by τ . Define $F_c : \text{ls}\mathfrak{c}_m \rightarrow \text{ls}\mathfrak{c}_m$ by setting $F_c(\rho)(s, s')$ equal to*

$$\max_{a \in A} \left[\theta_a^c(s, s') + c \cdot \mathcal{K}(\rho)(P_a(s), P_a(s')) \right]$$

Then F_c has a unique 1-bounded fixed-point pseudometric $\rho_c^ \in \mathbb{C}^b(S \times S)$ whose kernel is bisimilarity.*

We call such a metric a *bisimilarity metric*, and more generally a *bisimulation metric* if its kernel is a bisimulation relation (but not necessarily the largest). The following result, Theorem 3.20 in (Ferns et al., 2011), relates the optimal values of states to their similarity as measured by the bisimulation metric.

Theorem 2.6. *Assume the setup and result of Theorem 2.5. Let $\gamma \in (0, c]$ be a reward discount factor and let V_γ^* be the optimal value function defined in Theorem 2.1. Then V_γ^* is Lipschitz continuous with respect to ρ_c^* with Lipschitz constant $(1 - c)^{-1}$, i.e., for all $s, s' \in S$,*

$$|V_\gamma^*(s) - V_\gamma^*(s')| \leq (1 - c)^{-1} \rho_c^*(s, s')$$

Since bisimulation is a behavioural equivalence, this result implies that the closer two states are in bisimilarity distance, the more likely they are to share the same optimal actions, and hence optimal policies, for achieving the same optimal values.

3 A BISIMULATION VALUE FUNCTION

Let $\mathcal{M} = (S, \mathcal{B}_S, A, (P_a)_{a \in A}, r)$ be an MDP with the image of r contained in $[0, 1]$ and let $c \in (0, 1)$ be a discount factor. The goal of this section is to show that the bisimilarity metric ρ_c^* given by Theorem 2.5 can be expressed as the optimal value function of some MDP. In order to do so, let us first extend the definition of a coupling of two probability measures to a coupling of two labelled Markov kernels in the obvious way.

Definition 3. *Let (X, \mathcal{B}_X) and (Y, \mathcal{B}_Y) be standard Borel spaces, and let $(X \times Y, \mathcal{B}_X \otimes \mathcal{B}_Y)$ be the product space. Let I be an index set, and let $K = (K_i)_{i \in I}$, $L = (L_i)_{i \in I}$, and $M = (M_i)_{i \in I}$ be labelled Markov kernels on X , Y , and $X \times Y$, respectively. Then M is a coupling of K and L if and only if for each $i \in I$, $x \in X$, and $y \in Y$, $M_i(x, y)$ is a coupling of $K_i(x)$ and $L_i(y)$ in the sense of Definition 2. We denote the set of all couplings of K and L by $\Lambda(K, L)$.*

Recall that ρ_c^* is the unique solution to the fixed-point equation

$$\rho_c^*(x, y) = \max_{a \in A} [\theta_a^c(s, s') + c \cdot \mathcal{K}(\rho_c^*)(P_a(x), P_a(y))].$$

Here is the crucial fact: Theorem 2.4 remarkably not only provides a statement of duality for each Kantorovich linear program $\mathcal{K}(\rho_c^*)(P_a(x), P_a(y))$, but guarantees the existence of minimizers in the minimization linear program as well. Therefore, for every $a \in A$ and $x, y \in S$ there exists $\lambda_{axy} \in \Lambda(P_a(x), P_a(y))$ such that $\mathcal{K}(\rho_c^*)(P_a(x), P_a(y)) = \lambda_{axy}(\rho_c^*)$. Suppose for every $a \in A$ the map from $S \times S$ to $\mathbb{P}(S \times S)$ that sends (x, y) to λ_{axy} is measurable. Then ρ_c^* would satisfy the Bellman optimality equation defined in Theorem 2.1 for the optimal value function with discount factor c for the MDP $(S \times S, \mathcal{B}_{S \times S}, A, (\lambda_a)_{a \in A}, \theta^c)$ where $\lambda_a(x, y) = \lambda_{axy}$. Remark that such a $\lambda = (\lambda_a)_{a \in A}$ is a coupling of P with P , where $P = (P_a)_{a \in A}$. Thus, if we can find a measurable way of selecting the minimizers amongst all the Kantorovich linear programs appearing in the definition of ρ_c^* , we will have shown that the bisimilarity metric is actually the optimal value function of an MDP whose labelled Markov kernel is a coupling of two copies of the labelled Markov kernel of the original model. This is our goal.

3.1 MEASURABLE SELECTORS AND SECTIONS

The following results can be found in Section 1.4 of (Doberkat, 2007) and Section 5 of (Srivastava, 2008).

Let X and Y be sets. A *multifunction* from X to Y is a set-valued map $\mathcal{R} : X \rightarrow 2^Y$ such that for all $x \in X$, $\mathcal{R}(x)$ is a nonempty subset of Y . A multifunction \mathcal{R} from X to Y can equivalently be viewed as a relation between X and Y .

Given a multifunction \mathcal{R} between measurable spaces X and Y , one usually seeks to measurably select a member of $\mathcal{R}(x)$ for each $x \in X$. Here, we recount one way of doing so.

Let \mathcal{R} be a multifunction from X to Y , and let $G \subseteq Y$. The *weak inverse* of G with respect to \mathcal{R} is the set $\exists \mathcal{R}(G) = \{x \in X \mid \exists y \in G \text{ such that } (x, y) \in \mathcal{R}\} = \{x \in X \mid \mathcal{R}(x) \cap G \neq \emptyset\}$. The importance of the weak inverse lies in trying to utilise the property of measurability for a multifunction \mathcal{R} . A measurable function requires the preimage of every measurable set to be measurable. Here, we need only consider the preimages of compact sets.

Suppose \mathcal{R} is a multifunction between a measurable space X and a Polish space Y . Let $G \subseteq Y$. If $\exists \mathcal{R}(G)$ is measurable whenever G is compact then \mathcal{R} is called a \mathcal{C} -measurable relation on $X \times Y$.

Assume X is a measurable space, Y is Polish, \mathcal{R} is a multifunction from X to Y and for each $x \in X$, $\mathcal{R}(x)$ is a non-empty closed subset of Y . Then a measurable map $f : X \rightarrow Y$ is called a *measurable selector* for \mathcal{R} if and only if $f(x) \in \mathcal{R}(x)$ for all $x \in X$.

The following measurable selection result can be found as Proposition 1.57 and Proposition 1.58 in (Doberkat, 2007).

Proposition 3.1. *Assume that X is a measurable space, Y is a Polish space, and R is a \mathcal{C} -measurable relation on $X \times Y$. Then there exists a measurable selector f for \mathcal{R} .*

Finally, the following appears in Proposition 2.34 of (Folland, 1999), and will be used in conjunction with the preceding measurable selection theorem to establish our main result.

Proposition 3.2. *Suppose that (X, \mathcal{B}_X) , (Y, \mathcal{B}_Y) , and (Z, \mathcal{B}_Z) are measurable spaces and that $f : X \times Y \rightarrow Z$ is a product-measurable function. Let $x \in X$. Define the X -section of f at x , $f_x : Y \rightarrow Z$, to be the function defined by $f_x(y) = f(x, y)$ for all $y \in Y$. Then f_x is measurable.*

3.2 BISIMILARITY AS A VALUE FUNCTION

Theorem 3.3. *Let us assume the setup and result of Theorem 2.5. Let $K = (K_a)_{a \in A} \in \Lambda(P, P)$, where $P = (P_a)_{a \in A}$. Define the coupling of \mathcal{M} with itself through K to be the MDP $\mathcal{M}(K) = (S \times S, \mathcal{B}_{S \times S}, A, (K_a)_{a \in A}, \theta^c)$. Let $V_c^*(K)$ denote its optimal value function with respect to c , as defined in Theorem 2.1. Then there exists a $K^* \in \Lambda(P, P)$ such that $\rho_c^* = V_c^*(K^*) = \min_{K \in \Lambda(P, P)} V_c^*(K)$.*

In order to prove Theorem 3.3, we will need to make use of the following result, which can be found within the proof of Lemma 3.14 in (Ferns et al., 2011).

Lemma 3.4. *Assume the setup and result of Theorem 2.5. Then for each $a \in A$, the map sending (s, s') to $\mathcal{K}(\rho_c^*)(P_a(s), P_a(s'))$ is continuous on $S \times S$.*

Proof of Theorem 3.3. In order to prove the existence of K^* , we follow the method of part 3 of the proof of Lemma 4.9 in (Doberkat, 2007). First, however, we appeal to Theorem 2.5 to assert the existence of a Polish topology τ on S making each r_a continuous and each P_a weakly continuous for all $a \in A$. Let $X = A \times S \times S$ and $Y = \mathbb{P}(S \times S)$. The set A is a Polish space since it is finite, and X is a Polish space since it is a finite product of Polish spaces. Additionally, Y is also a Polish space (Giry, 1982). Define $\mathcal{R} : X \rightarrow 2^Y$ by setting $\mathcal{R}(a, x, y)$ to be the set of all $\lambda \in \Lambda(P_a(x), P_a(y))$ such that $\mathcal{K}(\rho_c^*)(P_a(x), P_a(y)) = \lambda(\rho_c^*)$. Theorem 2.4 implies that each $\mathcal{R}(a, x, y)$ is non-empty. Suppose $(\lambda_n)_{n \in \mathbb{N}} \subseteq \mathcal{R}(a, x, y)$ converges to $\lambda \in$

Y . By Lemma 2.3, $\lambda \in \Lambda(P_a(x), P_a(y))$. Theorem 2.5 implies that $\rho_c^* \in \mathbb{C}^b(S \times S)$, so that by weak convergence $\lambda(\rho_c^*) = \lim_{n \rightarrow \infty} \lambda_n(\rho_c^*) = \mathcal{K}(\rho_c^*)(P_a(x), P_a(y))$. Therefore, $\lambda \in \mathcal{R}(a, x, y)$, i.e. $\mathcal{R}(a, x, y)$ is closed.

Next, let $G \subseteq Y$ be compact, hence, closed. We will show that $\exists \mathcal{R}(G)$ is closed, and hence measurable. Let $(a_n, x_n, y_n)_{n \in \mathbb{N}} \subseteq \exists \mathcal{R}(G)$ be a sequence converging to some $(a, x, y) \in X$. So there exists $(\lambda_n)_{n \in \mathbb{N}} \subseteq G$ such that $\lambda_n \in \mathcal{R}(a_n, x_n, y_n)$ for all $n \in \mathbb{N}$. Since G is compact, it is also sequentially compact and therefore there exists a subsequence $(\lambda_{k_n})_{n \in \mathbb{N}}$ converging to some $\lambda \in G$. Remark that since A is finite, the sequence $(a_n)_{n \in \mathbb{N}}$ is eventually constant, i.e. there exists $N \in \mathbb{N}$ such that $(a_n, x_n, y_n) = (a, x_n, y_n)$ for all $n \geq N$. Let $(\varphi, \psi) \in \mathbb{C}^b(S) \times \mathbb{C}^b(S)$. Then

$$\begin{aligned} & \int_{S \times S} [\varphi(s) + \psi(s')] \lambda(ds, ds') \\ &= \lim_{n \rightarrow \infty} \left(\int_{S \times S} [\varphi(s) + \psi(s')] \lambda_{k_n}(ds, ds') \right) \\ &= \lim_{n \rightarrow \infty} \left(\int_S \varphi(s) P_{a_{k_n}}(x_{k_n})(ds) \right. \\ & \quad \left. + \int_S \psi(s') P_{a_{k_n}}(y_{k_n})(ds') \right) \\ &= \lim_{n \rightarrow \infty} \left(\int_S \varphi(s) P_a(x_{k_n})(ds) \right. \\ & \quad \left. + \int_S \psi(s') P_a(y_{k_n})(ds') \right) \\ &= \int_S \varphi(s) P_a(x)(ds) + \int_S \psi(s') P_a(y)(ds') \end{aligned}$$

so that $\lambda \in \Lambda(P_a(x), P_a(y))$. Here we have used the weak convergence of $(\lambda_{k_n})_{n \in \mathbb{N}}$ to λ , $(P_a(x_{k_n}))_{n \in \mathbb{N}}$ to $P_a(x)$, and of $(P_a(y_{k_n}))_{n \in \mathbb{N}}$ to $P_a(y)$, and the repeated use of Lemma 2.2. Moreover, by weak convergence and Lemma 3.4

$$\begin{aligned} \lambda(\rho_c^*) &= \lim_{n \rightarrow \infty} \lambda_{k_n}(\rho_c^*) \\ &= \lim_{n \rightarrow \infty} \mathcal{K}(\rho_c^*)(P_a(x_{k_n}), P_a(y_{k_n})) \\ &= \mathcal{K}(\rho_c^*)(P_a(x), P_a(y)), \end{aligned}$$

whence it follows that $\lambda \in \mathcal{R}(a, x, y)$. Therefore, $\mathcal{R}(a, x, y) \cap G \neq \emptyset$, $(a, x, y) \in \exists \mathcal{R}(G)$, and $\exists \mathcal{R}(G)$ is closed and hence measurable. By definition, \mathcal{R} is a \mathcal{C} -measurable relation on $X \times Y$. Applying Proposition 3.1 there exists a measurable selector $f : X \rightarrow Y$ for \mathcal{R} . Finally, set $K^* = (K_a^*)_{a \in A}$ where $K_a^*(x, y) = f(a, x, y) \in \mathcal{R}(a, x, y)$ for all $a \in A$, $x, y \in S$. For each $a \in A$, K_a^* is simply the A -section of f at a , so that by Proposition 3.2, each $K_a^* \in \llbracket S \times S \rightarrow \mathbb{P}(S \times S) \rrbracket$. Therefore, $\rho_c^* = V_c^*(K^*)$, the optimal value function for $\mathcal{M}(K^*)$.

Clearly $\inf_{K \in \Lambda(P, P)} V_c^*(K) \leq V_c^*(K^*)$. To establish the reverse inequality, let $K = (K_a)_{a \in A} \in \Lambda(P, P)$. Then for

any $a \in A$ and $x, y \in S$,

$$\begin{aligned} & \theta_a^c(x, y) + c \cdot K_a^*(x, y)(\rho_c^*) \\ &= \theta_a^c(x, y) + c \cdot \mathcal{K}(\rho_c^*)(P_a(x), P_a(y)) \\ &= \theta_a^c(x, y) + c \cdot \inf_{\lambda \in \Lambda(P_a(x), P_a(y))} \lambda(\rho_c^*) \\ &\leq \theta_a^c(x, y) + c \cdot K_a(x, y)(\rho_c^*). \end{aligned}$$

By taking the maximum over all $a \in A$ and noting that the result holds for all $x, y \in S$, we then obtain $\rho_c^* \leq T_c(K)(\rho_c^*)$, where $T_c(K)$ is the Bellman optimality operator for the MDP $\mathcal{M}(K)$. Therefore, it follows that $V_c^*(K^*) \leq V_c^*(K)$ for any $K \in \Lambda(P, P)$, and finally that $V_c^*(K^*) \leq \inf_{K \in \Lambda(P, P)} V_c^*(K)$. \square

Thus, we can interpret every discounted bisimulation metric as the optimal value function of some MDP; moreover, that MDP is optimal in the sense that it is the best coupling of the transition structure of the original MDP with itself when one seeks to minimize the expected total discounted *absolute difference* in rewards coming from the original model.

An immediate consequence is that we can now interpret the topology of convergence with respect to a bisimulation metric in terms of MDP optimality criteria. Conversely, when examining behavioural equivalence for the state space of a given MDP it no longer suffices to consider the structural model alone; one *must* take into account the full Markov decision problem, i.e. its intended use by means of an optimality criterion. This is yet another advantage of the pseudometric approach over that of exact equivalences. We discuss this further in Section ??.

Practical implications are less immediate, but no less important, particularly in regard to determining what might be effective in attempting to calculate or estimate the distances. Consider a finite MDP. If we adjoin one new absorbing state with no immediate rewards then it is not hard to show that the bisimulation distance from that state to another state is the optimal value of the latter state. So computing a bisimulation metric is at least as hard as computing an optimal value function. On the other hand, we have just shown that computing a bisimulation metric amounts to computing an optimal value function - albeit, with the caveat that this amounts to a search over possibly infinitely many couplings. If one could restrict this search to polynomially many couplings, then it would follow that computing bisimulation metrics and computing optimal value functions belong to the same polynomial-time complexity class - and we conjecture that this is so. At first glance, this is a disappointing result; if one followed the naive approach, one would be attempting to solve for an optimal value function by solving for another optimal value function over a quadratically larger MDP. However, computing a bisimulation metric is of interest in its own right; the value function formulation allows for state-of-the-art

reinforcement learning techniques (Sutton & Barto, 2012; Pazis & Parr, 2013) to be applied in its computation while at the same time informing us of what methods are unlikely to work well in practice for truly large systems.

A better practical approach would be to find more easily computable similarity metrics that are related in some meaningful way to the bisimulation metric. In that case, one would have a practical similarity measure with the theoretical guarantees given by bisimulation, as in Theorem 2.6. Our value function formulation permits a very natural way to do this, through the use of couplings.

Definition 4. Let $f : X \times X \rightarrow [0, \infty)$ be a function on a set X . Define the function $\varrho(f) : X \times X \rightarrow [0, \infty)$ by $\varrho(f)(x, y) = \inf\{\sum_{j=1}^m \omega(f)(a_{j-1}, a_j)\}$, where $\omega(f)(u, v) = \min\{f(u, v), f(v, u)\}$ and the infimum is taken over all $m \in \mathbb{N}$ and $(a_j)_{j=0}^m \subseteq X$ such that $a_0 = x$ and $a_m = y$. Then $\varrho(f)$ is the largest pseudometric less than f .

Notice that if X is finite and f is computable then the problem of computing $\varrho(f)$ is the *All-Pairs Shortest Paths* problem.

Corollary 3.5. Assume the setup and result of Theorem 3.3. For $\pi \in \Pi$ defined over $S \times S$ we let $V_c(K)(\pi)$ denote the value function of $\mathcal{M}(K)$ with respect to π and c and we let $V_c^*(K)$ denote its optimal value function with respect to c , as defined in Theorem 2.1. Then

1. $\forall K \in \Lambda(P, P), \rho_c^* \leq \varrho(V_c^*(K)) \leq V_c^*(K)$.
2. $\forall \pi \in \Pi, \varrho(V_c(K^*)(\pi)) \leq V_c(K^*)(\pi) \leq \rho_c^*$.

Corollary 3.5 allows us to easily bound the bisimulation metric from above for any coupling $K \in \Lambda(P, P)$. For example, the product coupling $P \otimes P$ defined in the obvious way (and assuming measurability) by $(P \otimes P)_a(x, y) = P_a(x) \otimes P_a(y)$ should provide a trivial upper bound. Corollary 3.5 also provides a lower bound but only in the case where we know the optimal coupling K^* beforehand. Potentially more interesting is the case where we combine the two, i.e. for an arbitrary coupling $K \in \Lambda(P, P)$ and an arbitrary policy $\pi \in \Pi$ defined on $\mathcal{M}(K)$, does the equivalence induced by $\varrho(V_c(K)(\pi))$ lead to something that is more easily computable but that still provides good theoretical guarantees?

4 RELATED WORK

This work lies at the intersection of artificial intelligence and formal verification, and owes much to both. The concept of bisimulation has been in use within the uncertainty in artificial intelligence community for some time now. Indirectly in (Boutilier et al., 2000) and directly in (Givan et al., 2003), the notion of bisimulation had

been transferred from the theory of concurrent processes to MDP model minimization and the reinforcement learning paradigm. These papers work directly with factored or structured representations, which is an advantage over our approach for problems where such structure in the environment exists and is known explicitly. On the other hand, they deal only with discrete MDPs, exhibit the brittleness inherent in using exact equivalences for numerical systems, and lack theoretical guarantees on the size of a fully minimal system. In earlier work, Dean et al. (1997) actually consider an approximate version of bisimulation. For a small positive parameter ε they consider equivalence relations satisfying the property that immediate rewards and stochastic transitions to equivalence classes differ by at most ε . However, the disadvantages already mentioned still apply. More generally, (Li et al., 2006) provide a comprehensive survey and classification of various state abstractions for finite MDPs, including methods based on bisimulation (such as our bisimulation metrics).

Bisimulation metrics have been more extensively studied in the formal verification community. In that setting, the work closest in spirit to our own is (Chen et al., 2012), wherein the authors investigate the complexity of computing bisimilarity and metric bisimilarity for labelled Markov chains. In particular, Theorem 8 in that work relates an undiscounted bisimulation metric to optimal couplings of a given labelled Markov chain. Aside from considering only finite state systems, they allow for states to have differing sets of permissible actions but omit the reward parameter; hence, their work lies outside of the optimal control theory framework on which we focus.

Abate (2012) surveys various approximation metrics for probabilistic bisimulation over Markov processes with general state and action spaces, though here too Markov reward processes are mostly neglected. The author does conclude that a bridge needs to be made between techniques based on computing distances between Markov kernels and techniques based on sampling trajectories from processes under consideration; we believe the current work can help provide that bridge.

A very promising approach appears in (Desharnais et al., 2013) where the authors propose a general algorithm for estimating divergences, distance functions that may fail to satisfy the symmetry and triangle inequality axioms of a pseudometric. They consider divergences that generalize equivalences on probabilistic systems based on tests and observations. In particular, they define a new family of testable equivalences called *k-moment equivalences*; 1-moment equivalence is trace equivalence, as k grows larger k -moment equivalence becomes finer, and all k -moment equivalences as well as their limit equivalence are strictly weaker than bisimilarity. The exciting feature of their work is that the algorithm for estimating a divergence corresponding to a fixed equivalence is based on defining an

MDP whose optimal value function is that divergence, and then using reinforcement learning techniques (Sutton & Barto, 2012) to solve for the optimal value function. While conceptually similar in spirit to our value function representation of bisimulation metrics, this approach differs significantly in how the MDP representing the metric is defined.

5 CONCLUSIONS AND FUTURE WORK

We have shown that the bisimulation metric defined in (Ferns et al., 2011) for an MDP is actually the optimal value function of an optimal coupling of the MDP with itself. This latter formulation is perhaps a more natural conception of distance for MDPs. In any case, all theoretical and practical results from optimal control theory concerning optimal value functions for MDPs can be carried over (based on this result) to the study of bisimulation metrics.

Perhaps the most intriguing implication of Theorem 3.3 is that examining other optimal control theory criteria may lead to different classes of bisimulation metrics perhaps better suited to those optimality tasks. Consider the undiscounted case. What does ρ_c^* represent when c tends to 1? We could set $c = 1$ in Theorem 2.5, as in Theorem 4 of (Chen et al., 2012). The resultant functional $F_1 : \mathfrak{Lsc}_m \rightarrow \mathfrak{Lsc}_m$ defined by $F_1(\rho)(s, s') = \max_{a \in A} K(\rho)(P_a(s), P_a(s'))$ has a least fixed-point ρ_1^* given by the Knaster-Tarski fixed-point theorem. In fact, in this case the least fixed-point is the everywhere zero pseudometric - unsurprising since in our current setup all actions are allowable in all states and the reward parameter is the only feature that distinguishes states. But how might we interpret such a result more generally? In fact, there is some relation to the infinite-horizon average reward optimality criterion.

Let $\mathcal{M} = (S, \mathcal{B}_S, A, (P_a)_{a \in A}, r)$ be an MDP with the image of r contained in $[0, 1]$ and recall the terminology of Section 2.2. The following definitions can be found in Chapter 5 of (Hernández-Lerma & Lasserre, 1996). Let $\pi \in \Pi$ be a policy on \mathcal{M} . Let $n \in \mathbb{N}$. The *n-stage value function* for π is defined by $J_n(\pi)(s) = \mathbb{E}_s^\pi[\sum_{t=0}^{n-1} r(a_t, x_t)]$ for all $s \in S$, the *average cost value function* for π by $J(\pi)(s) = \limsup_{n \rightarrow \infty} \frac{1}{n} J_n(\pi)(s)$ for all $s \in S$, and the average reward optimal value function by $J^*(s) = \sup_{\pi \in \Pi} J(\pi)(s)$ for all $s \in S$. The solution to the average reward Markov decision problem is a policy π^* such that $J(\pi^*) = J^*$.

Let us assume that \mathcal{M} is finite, i.e., S is finite. The following can be found in Chapter 8 of (Feinberg & Shwartz, 2002), in particular Theorem 8.1, listed below as Theorem 5.1. A policy $\pi \in \Pi$ is said to be *Blackwell optimal* if and only if there exists $\gamma_0 \in (0, 1)$ such that π is γ -optimal for all $\gamma \in (\gamma_0, 1)$.

Remark 2. A stationary Blackwell optimality policy is also average reward optimal, and for such a policy π^* , $\lim_{\gamma \uparrow 1} (1 - \gamma)V_\gamma(\pi^*) = J(\pi^*)$.

Theorem 5.1. In a finite MDP there exists a stationary Blackwell optimal policy.

Let $K \in \Lambda(P, P)$ and $V_c^*(K)$ be the optimal value function for the MDP $\mathcal{M}(K)$ defined in Section 3 with the reward parameter θ^c scaled by $(1 - c)^{-1}$. Then there exists a $K_c^* \in \Lambda(P, P)$, depending on c , such that $\rho_c^* = (1 - c)V_c^*(K_c^*)$. It follows that $\lim_{c \uparrow 1} \rho_c^* = \lim_{c \uparrow 1} (1 - c)V_c^*(K_c^*) \leq \lim_{c \uparrow 1} (1 - c)V_c^*(K) \leq J^*(K)$ for any $K \in \Lambda(P, P)$. Thus, $\lim_{c \uparrow 1} \rho_c^* \leq \inf_{K \in \Lambda(P, P)} J^*(K)$. It remains to be seen whether or not the inequality is strict.

In the general case, the situation is much more complicated. For example, under a variety of conditions not listed here, Lemma 10.4.3 of (Hernández-Lerma & Lasserre, 1999) states the following.

Lemma 5.2. There exists a constant α such that $\alpha = \limsup_{\gamma \uparrow 1} (1 - \gamma)V_\gamma^* \leq J^*$.

It follows that under the same conditions $\limsup_{c \uparrow 1} \rho_c^* \leq \alpha \leq \inf_{K \in \Lambda(P, P)} J^*(K)$. If equality were to hold in this case, then we would have for some $x \in S$, $\alpha = \limsup_{c \uparrow 1} \rho_c^*(x, x) = 0$, so that $\lim_{c \uparrow 1} \rho_c^*$ exists and is everywhere zero, i.e. the resulting equivalence would identify all states. Aside from the unspecified conditions, the distinction with the finite case is that $\lim_{c \uparrow 1} \rho_c^*$ need not exist to begin with.

Similarly, consider the expected total reward criterion. Here we might take the set of lower semicontinuous pseudometrics on S as in Theorem 2.5, but this time bounded with respect to a weighted supremum norm $\|\cdot\|_w$ for some weight function $w : S \times S \rightarrow [1, \infty)$. Thus, an unbounded function f that has a bounded norm with respect to w has its rate of growth bounded by w . Define the functional F by

$$F(\rho)(s, s') = \max_{a \in A} [\theta_a^0(s, s') + K(\rho)(P_a(s), P_a(s'))].$$

Then if conditions are imposed so that the set of w -bounded lower semicontinuous pseudometrics on S is closed under F , it will have a least fixed-point (extended) pseudometric again corresponding to some expected total reward optimal value function. This line of research is very preliminary.

The major concern of this work along with (Ferns et al., 2014) is to clarify and unify results about the theory of bisimulation metrics in order to provide new avenues of attack for practical applications. An ongoing research goal is to find a more easily computable equivalence than that given by the current bisimulation metrics while maintaining as much as possible the theoretical guarantees. As far as estimating the given family of bisimulation metrics, the current interpretation as optimal value functions suggests

that the most promising approaches involve Monte Carlo techniques, as in (Ferns et al., 2006), (Ferns et al., 2011), and most recently in (Comanici et al., 2012), or advanced approximate linear programming techniques as in (Pazis & Parr, 2013). More to the point, our strong intuition is that state-of-the-art methods for efficient reinforcement learning can be leveraged to develop state-of-the-art methods for efficient bisimulation metric computation, and vice versa. Very interesting recent work in this direction have been done by Bacci et al. (2013), who use greedy heuristics and decomposition techniques to speed up the computation of bisimulation metrics for MDPs. Computational approaches of this flavour should be further investigated.

In order for this approach to be really useful in practice, however, a few topics need to be further addressed by future work.

First, this work is highly dependent on couplings and the coupling method, though we have only just touched upon the subject. The study of couplings in theory and in practice is vast, and a proper discussion is beyond the scope of this work. A good source for the theory of couplings is (Lindvall, 2002). Moreover, as noted in (Chen et al., 2012), it is already known in the discrete case that the set of couplings (called *matchings* in that work) for two probability functions forms a polytope; and that optimizing a linear function over it amounts to optimizing over the finitely many vertices of the polytope (as is done in computing the discrete Kantorovich metric). We hope that this structure can be exploited to improve our theoretical result.

Additionally, we mentioned that the initial applications of bisimulation to MDPs exploited factored or structured representations. It would be fruitful to explore whether or not bisimulation metric reasoning principles can be applied to factored representations without having to flatten the state space. More generally, applying bisimulation metrics to the problem of constructing function approximators for MDP value functions is a very promising future direction, recent work (Comanici & Precup, 2012) has leveraged such metrics to tackle the problem of automatically generating features for function approximation.

Lastly, let us consider the problem of knowledge transfer between MDPs. Suppose $\mathcal{M}_X = (X, \mathcal{B}_X, A, P_X, r(X))$ and $\mathcal{M}_Y = (Y, \mathcal{B}_Y, A, P_Y, r(Y))$ are two MDPs with rewards in the unit interval and that $c \in (0, 1)$ is a discount factor. For $K \in \Lambda(P_X, P_Y)$, consider the coupled MDP $\mathcal{M} = (X \times Y, \mathcal{B}_{X \times Y}, A, K, \theta)$ where $\theta_a(x, y) = (1 - c)|r_a(X)(x) - r_a(Y)(y)|$ for all $x \in X$ and $y \in Y$. Does $\inf_{K \in \Lambda(P_X, P_Y)} V_c^*(K)(x, y)$ measure the bisimilarity of states x and y ? Clearly, there is much work to be done to answer this question.

Acknowledgements

This work is dedicated with love to Prakash Panangaden in honour of his 60th birthday.

References

- Abate, A. (2012). Approximation Metrics based on Probabilistic Bisimulations for General State-Space Markov Processes: a Survey. *Electronic Notes in Theoretical Computer Sciences*.
- Bacci, G., Bacci, G., Larsen, K. G., & Mardare, R. (2013). Computing behavioral distances, compositionally. *Proceedings of the 38th International Symposium on Mathematical Foundations of Computer Science (MFCS)* (pp. 74–85).
- Boutillier, C., Dearden, R., & Goldszmidt, M. (2000). Stochastic Dynamic Programming with Factored Representations. *Artificial Intelligence*, 121, 49–107.
- Chen, D., van Breugel, F., & Worrell, J. (2012). On the Complexity of Computing Probabilistic Bisimilarity. *FoSSaCS* (pp. 437–451). Springer.
- Comanici, G., Panangaden, P., & Precup, D. (2012). On-the-Fly Algorithms for Bisimulation Metrics. *QEST* (pp. 94–103). IEEE Computer Society.
- Comanici, G., & Precup, D. (2012). Basis Function Discovery Using Spectral Clustering and Bisimulation Metrics. *AAAI*.
- Dean, T., Givan, R., & Leach, S. (1997). Model Reduction Techniques for Computing Approximately Optimal Solutions for Markov Decision Processes. *UAI* (pp. 124–131).
- Desharnais, J., Jagadeesan, R., Gupta, V., & Panangaden, P. (2002). The Metric Analogue of Weak Bisimulation for Probabilistic Processes. *LICS* (pp. 413–422). IEEE Computer Society.
- Desharnais, J., Laviolette, F., & Zhioua, S. (2013). Testing Probabilistic Equivalence Through Reinforcement Learning. *Information and Computation*, 227, 21–57.
- Doberkat, E.-E. (2007). *Stochastic Relations. Foundations for Markov Transition Systems*. Chapman & Hall/CRC.
- Feinberg, E., & Schwartz, A. (Eds.). (2002). *Handbook of Markov Decision Processes - Methods and Applications*. Kluwer International Series.
- Ferns, N., Castro, P. S., Precup, D., & Panangaden, P. (2006). Methods for Computing State Similarity in Markov Decision Processes. *UAI*.
- Ferns, N., Panangaden, P., & Precup, D. (2004). Metrics for Finite Markov Decision Processes. *UAI* (pp. 162–169).
- Ferns, N., Panangaden, P., & Precup, D. (2005). Metrics for Markov Decision Processes with Infinite State Spaces. *UAI* (pp. 201–208).
- Ferns, N., Panangaden, P., & Precup, D. (2011). Bisimulation Metrics for Continuous Markov Decision Processes. *SIAM Journal on Computing*, 40, 1662–1714.
- Ferns, N., Precup, D., & Knight, S. (2014). Bisimulation for Markov Decision Processes Through Families of Functional Expressions. *Horizons of the Mind. A Tribute to Prakash Panangaden* (pp. 319–342). Springer.
- Folland, G. B. (1999). *Real analysis: Modern techniques and their applications*. Wiley-Interscience. Second edition.
- Giry, M. (1982). A Categorical Approach to Probability Theory. *Categorical Aspects of Topology and Analysis*, 68–85.
- Givan, R., Dean, T., & Greig, M. (2003). Equivalence Notions and Model Minimization in Markov Decision Processes. *Artificial Intelligence*, 147, 163–223.
- Hernández-Lerma, O., & Lasserre, J. B. (1996). *Discrete-Time Markov Control Processes : Basic Optimality Criteria*. Applications of Mathematics. Springer.
- Hernández-Lerma, O., & Lasserre, J. B. (1999). *Further Topics on Discrete-Time Markov Control Processes*. Applications of Mathematics. Springer.
- Larsen, K. G., & Skou, A. (1991). Bisimulation Through Probabilistic Testing. *Information and Computation*, 94, 1–28.
- Li, L., Walsh, T. J., & Littman, M. L. (2006). Towards a Unified Theory of State Abstraction for MDPs. *Proceedings of the Ninth International Symposium on Artificial Intelligence and Mathematics* (pp. 531–539).
- Lindvall, T. (2002). *Lectures on the Coupling Method*. Dover Publications Inc.
- Pazis, J., & Parr, R. (2013). Sample Complexity and Performance Bounds for Non-Parametric Approximate Linear Programming. *AAAI*.
- Puterman, M. L. (1994). *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons, Inc.
- Srivastava, S. M. (2008). *A Course on Borel Sets*, vol. 180 of *Graduate texts in mathematics*. Springer.
- Sutton, R. S., & Barto, A. G. (2012). *Reinforcement Learning: An Introduction (Second Edition, In Progress)*. MIT Press.

- van Breugel, F., & Worrell, J. (2001a). Towards Quantitative Verification of Probabilistic Transition Systems. *ICALP* (pp. 421–432). Springer.
- Villani, C. (2003). *Topics in Optimal Transportation* (*Graduate Studies in Mathematics, Vol. 58*). American Mathematical Society.

Annealing Paths for the Evaluation of Topic Models

James Foulds Padhraic Smyth

Department of Computer Science
University of California, Irvine
Irvine, CA 92697, USA
{jfoulds, smyth}@ics.uci.edu

Abstract

Statistical topic models such as latent Dirichlet allocation have become enormously popular in the past decade, with dozens of learning algorithms and extensions being proposed each year. As these models and algorithms continue to be developed, it becomes increasingly important to evaluate them relative to previous techniques. However, evaluating the predictive performance of a topic model is a computationally difficult task. Annealed importance sampling (AIS), a Monte Carlo technique which operates by annealing between two distributions, has previously been successfully used for topic model evaluation (Wallach et al., 2009b). This technique estimates the likelihood of a held-out document by simulating an annealing process from the prior to the posterior for the latent topic assignments, and using this simulation as an importance sampling proposal distribution.

In this paper we introduce new AIS annealing paths which instead anneal *from one topic model to another*, thereby estimating the *relative* performance of the models. This strategy can exhibit much lower empirical variance than previous approaches, facilitating reliable per-document comparisons of topic models. We then show how to use these paths to evaluate the predictive performance of topic model learning algorithms by efficiently estimating the likelihood at each iteration of the training procedure. The proposed method achieves better held-out likelihood estimates for this task than previous algorithms with, in some cases, an order of magnitude less computation.

1 INTRODUCTION

Topic models such as latent Dirichlet allocation (Blei et al., 2003) have become standard tools for analyzing text cor-

pora, with broad applications in areas such as political science (Grimmer, 2010), sociology (McFarland et al., 2013), conversational dialog (Nguyen et al., 2013), and more. A multitude of extensions to the LDA model have been developed for finding meaningful latent structure in text, along with a variety of strategies for fitting these models to increasingly large corpora.

As these new ideas continue to be proposed in the literature it becomes increasingly important to obtain accurate quantitative evaluations of the different approaches. Among the techniques available for evaluating topic models, the prediction of words in held-out documents (via test log-likelihood or perplexity) is perhaps the single most widely-used method for benchmarking the performance of new topic models and inference algorithms. An important point is that speedups for training these models do not necessarily translate to speedups in evaluating them. For example, there now exist very fast learning algorithms for training topic models based on approximate inference techniques, such as stochastic variational inference (Hoffman et al., 2010, 2013; Foulds et al., 2013), making it possible to learn topic models on corpora with millions of documents. Ironically, however, the time taken to compute test-set metrics for these algorithms can be orders of magnitude greater than the time it takes to train them. The evaluation of the predictive performance of topic models on held-out documents is still painfully slow, and relatively unreliable for individual documents as we will see later in the paper.

More specifically, consider a held-out document d , with word vector $w^{(d)}$, in the context of evaluating the quality of an LDA topic model (or one its many extensions). Given point estimates of topics Φ and a potentially document-specific Dirichlet prior α (if learned), we wish to compute the likelihood of the words in this held-out document, $Pr(w^{(d)}|\Phi, \alpha)$.¹ The direct computation of this quantity involves either an intractable sum over the latent topic assignments $z^{(d)}$, or an intractable integral over the distribution over topics $\theta^{(d)}$. Moreover, this already difficult computation must be performed for every document in

¹Or perplexity, a function of this and document length.

the held-out test set, which frequently contains hundreds to thousands of documents. To address this challenge, a wide variety of approximation strategies for estimating $Pr(w^{(d)}|\Phi, \alpha)$ have been proposed in papers such as those from Wallach et al. (2009b), Buntine (2009) and Scott & Baldridge (2013). Although these methods can lead to significantly more accurate results than naive approaches, the reliable and efficient evaluation of topic models remains a relatively open problem of practical significance.

In this paper we investigate new methods for evaluating topic models based on annealed importance sampling (AIS) (Neal, 2001), a Monte Carlo integration technique which was previously applied to topic model evaluation by Wallach et al. (2009b). Given two probability distributions, AIS produces an estimate of the ratio of their partition functions by annealing between them. Wallach et al. leverage this idea by annealing from the prior over the latent topic assignments $z^{(d)}$ to the posterior, resulting in an estimate of held-out document likelihood. AIS can be very accurate given enough computation time, although the amount of time needed may vary greatly between different choices of annealing paths (Grosse et al., 2013).

The first contribution of this paper is to propose and evaluate an alternative annealing strategy, using two AIS paths which anneal from one topic model to another. This strategy (referred to as ratio-AIS) computes the *ratio* of the likelihoods of two models instead of computing the likelihoods of each model separately. The result is an estimate of the relative performance of the models, with significantly lower empirical variance across runs than previous approaches.² This in turn brings computational benefits, as fewer samples or annealing temperatures may be required to achieve reliable results. The reduced variance comes at the cost of potentially increased bias when insufficient iterations are performed to achieve convergence. However, we also show how to detect such bias by annealing between the topic models in both directions and comparing the results. The consequence of this bias-variance trade-off is that the proposed method is useful in cases where we would like to perform in-depth analysis at the per-document level and when the two topic models are similar to each other. The previous high-variance low-bias methods may still be preferred for general full-corpus comparisons of topic models.

Finally, we show how to use the proposed AIS paths for evaluating topic model learning algorithms by computing held-out likelihood curves over the iterations of the learning procedure. This is achieved by annealing between the topic models at each iteration of the learning algorithm in turn, which allows all previous computation to be reused in each of the likelihood estimates. The proposed method outperforms previous algorithms, in some cases even when

it is given an order of magnitude less computation time. Note that although we focus on topic models, the ideas presented here could potentially also be useful for other latent variable models with intractable likelihoods.

2 BACKGROUND

When proposing a new topic model or learning algorithm, it is important to evaluate its performance. When the model is to be used for a certain task it may be possible to evaluate it with respect to an extrinsic, task-specific metric. For example one could evaluate the quality of topics being used as features for a classification algorithm by measuring classification accuracy. More generally, however, given that topic models are generally trained in an unsupervised manner (with a few notable exceptions), a ground-truth evaluation metric is typically not available.

Consequently, a number of intrinsic (i.e. task independent) validation strategies for topic models have been developed in the literature. For example, Chang et al. (2009) proposed the use of elicitation of judgments from humans to evaluate the quality of topic models. Given that obtaining these judgments can be expensive and difficult, Newman et al. (2010) and Mimno et al. (2011) proposed automatic surrogate measures of topic coherence, and showed that these measures, which are typically based on word co-occurrence statistics, are correlated with human judgments.

However, as topic models are statistical models, we also would like to be able to evaluate them as such. In the context of unsupervised machine learning, the standard approach for evaluating a statistical model is to compute the probability of held-out data. Regardless of the utility of the aforementioned methods, it is generally useful to demonstrate good predictive performance in addition to any other extrinsic or intrinsic validation results. Intuitively, as our goal is to fit a statistical model to data, we would like to know both how well we are able to fit the model, and how well the model is able to explain unobserved data.

As in Wallach et al. (2009b), we therefore focus on the computation of $Pr(w^{(d)}|\Phi, \alpha)$, the likelihood of the words $w^{(d)}$ in a held-out document d (or equivalently, perplexity), conditioned on point estimates of the topic-word distributions Φ and (possibly document-specific) priors α , where Φ is a $W \times K$ matrix consisting of K discrete distributions $\Phi^{(k)}$ over the W words in the dictionary, and α is a K -dimensional Dirichlet parameter vector.³ This quantity can be used to evaluate a point estimate of the topics, or in an inner loop to evaluate Bayesian evaluation metrics such as the posterior predictive probability of held-out documents.

²“Variance” here refers to variance across Monte Carlo estimates of the difference in log-likelihood between models, per document.

³It is standard practice to learn an asymmetric Dirichlet prior α in LDA models, following Wallach et al. (2009a), so we include it as a parameter to evaluate. The prior may also be learned in a document dependent way for models such as DMR (Mimno & McCallum, 2008).

It is in general infeasible to compute $Pr(w^{(d)}|\Phi, \alpha)$ directly, as it involves an intractable sum $\sum_z^{(d)} Pr(w^{(d)}, z^{(d)}|\Phi, \alpha)$ or an intractable integral $\int_{\theta} Pr(w^{(d)}, \theta^{(d)}|\Phi, \alpha)$. The computational difficulty arises because the topic assignments $z^{(d)}$ and distributions over topics $\theta^{(d)}$ for the held-out document are unknown, and so all possible values must be considered. A variety of approximation strategies were considered by Wallach et al. (2009b), the number of which alone is a testament to the difficulty of the problem. The most widely used of these approaches is the “left-to-right” particle filtering algorithm. In the algorithm, a number of particles are maintained, representing topic assignments up to the current word t in the document. In each iteration, these particles are used to draw samples of the topic assignment for the next word $t + 1$, conditioned on the previous words and topic assignments. A resampling step is also performed, making the algorithm’s run time quadratic in the length of the document. The algorithm was analyzed more closely by Buntine (2009), and a faster, but less accurate, variant of the technique was proposed by Scott & Baldridge (2013).

Alternatively, a strategy for side-stepping some of the computational difficulty is to instead estimate (or sample) $z^{(d)}$ or $\theta^{(d)}$ on a subset $w^{(d,1)}$ of the document, and predict only the remaining portion of the document $w^{(d,2)}$, thus estimating $Pr(w^{(d,2)}|w^{(d,1)}, \Phi, \alpha)$. This method is frequently used in practice (e.g. Rosen-Zvi et al. (2004); Wallach et al. (2009a)). However, this “document completion” scenario changes the task somewhat, and is not the gold standard prediction task we would like it to be. It measures the ability of the model to “orient” itself quickly when given partial documents, rather than how likely the overall document is under the model. The widespread use of the document completion strategy may be largely due to its convenient computational properties (leading in turn to its use as a surrogate for fully held-out prediction), rather than being due to any intrinsic benefit of the metric itself.

It is also unclear how the use of document completion as a surrogate for full-document prediction might affect our conclusions, particularly when using topic models which learn the Dirichlet hyper-parameter α as in Wallach et al. (2009a) and Mimno & McCallum (2008). Learning α may help the model to recover $\theta^{(d)}$ better on the training portion of the document, thus increasing the performance of the model for document completion more than in the fully held-out case.

On the other hand, observing more of the document decreases the relative impact of the prior on the posterior distribution, which could reduce the observed improvement due to learning α . Thus, we suspect that document completion may not always be a good surrogate for the full prediction task. It should be noted that many methods for fully held-out prediction can also be adapted for document completion (including those proposed here).

2.1 ANNEALED IMPORTANCE SAMPLING

One of the more accurate strategies investigated by Wallach et al. (2009b) to estimate held-out likelihood was annealed importance sampling (AIS) (Neal, 2001). AIS is a general technique for estimating an expectation of a function of a random variable x with respect to an intractable distribution of interest p_0 . Consider a distribution p_n (which is typically easy to sample from) and a sequence of “intermediate” distributions p_{n-1}, \dots, p_1 leading from p_n to p_0 . AIS works by annealing from p_n towards p_0 by way of the intermediate distributions, and using importance weights to correct for the fact that an annealing process was used instead of sampling directly from p_0 .

Assume that for each intermediate distribution p_j we have a Markov chain with transition operator $T_j(x, x')$ which is invariant to that distribution. We need to be able to sample from these Markov chains, and for each p_j be able to evaluate some function f_j which is proportional to it. In a manner similar to that of traditional importance sampling, AIS produces a collection of samples $x^{(1)}, \dots, x^{(S)}$ with associated importance weights $w^{(1)}, \dots, w^{(S)}$. As with importance sampling, the expectation of interest is estimated using the samples, weighted by the importance weights.

The strategy for drawing each sample $x^{(i)}$ is to begin by drawing a sample x_{n-1} from p_n , then drawing a sequence of points x_{n-2}, \dots, x_0 which “anneal” towards p_0 . Each of the remaining x_j ’s in the sequence are generated from x_{j+1} via T_j . Importance weights $w^{(i)}$ are computed by viewing (x_0, \dots, x_{n-1}) as an augmented state space, and performing importance sampling on this new state space. The above procedure is used as a proposal distribution Q for importance sampling from another distribution P :

$$Q(x_0, \dots, x_{n-1}) \propto f_n(x_{n-1}) \prod_{j=n-1}^1 T_j(x_j, x_{j-1})$$

$$P(x_0, \dots, x_{n-1}) \propto f_0(x_0) \prod_{j=1}^{n-1} \tilde{T}_j(x_{j-1}, x_j),$$

where $\tilde{T}_j(x, x') = T_j(x', x) \frac{f_j(x')}{f_j(x)}$ is the reversal of the transition defined by T_j . This leads to importance weights for each of the samples,

$$w^{(i)} = \frac{P(x_0, \dots, x_{n-1})}{Q(x_0, \dots, x_{n-1})} = \prod_{j=0}^{n-1} \frac{f_j(x_j)}{f_{j+1}(x_j)}. \quad (1)$$

Note that the marginal probability of x_0 under P is $p_0(x_0)$, so after letting $x^{(i)} = x_0$ the procedure correctly carries out importance sampling from p_0 . AIS also provides an estimate for the ratio of normalizing constants for f_0 and f_n . The normalizing constant for P is the same as the normalizing constant for f_0 , and the normalizing constant for Q is the same as the normalizing constant for f_n , and so

the average of the importance weights, $\frac{\sum w^{(i)}}{N}$, converges to $\frac{\int f_0(x)dx}{\int f_n(x)dx}$.

2.2 AIS FOR TOPIC MODELS

Wallach et al. (2009b) showed how to apply the AIS procedure to the problem of calculating LDA likelihoods. The likelihood of a test document for a topic model can be estimated by using AIS to estimate a normalization constant, operating on the latent topic assignments $z^{(d)}$ for the document.⁴ We can set $f_0 = Pr(w^{(d)}, z^{(d)}|\Phi, \alpha)$, $f_n = Pr(z^{(d)}|\alpha)$, with intermediate distributions $f_j = Pr(w^{(d)}|z^{(d)}, \Phi, \alpha)^{\beta_j} f_n$ and the transition operators T_j being the Gibbs sampler for f_j . The ratio of normalizing constants is

$$\begin{aligned} \frac{\sum w^{(i)}}{S} &\approx \frac{\sum_{z^{(d)}} Pr(w^{(d)}, z^{(d)}|\Phi, \alpha)}{\sum_{z^{(d)}} Pr(z^{(d)}|\alpha)} \\ &= \frac{Pr(w^{(d)}|\Phi, \alpha)}{1} = Pr(w^{(d)}|\Phi, \alpha). \end{aligned} \quad (2)$$

The procedure for producing each importance sample, then, is to draw an initial $z^{(d)}$ from the prior, and anneal it towards f_0 by performing $r_j \geq 1$ Gibbs iterations at each intermediate distribution. After repeating this procedure for each sample, the likelihood is estimated as the average of the importance weights. Note that in what follows we define a *run* as the full procedure averaging over importance samples, while a *sample* refers to a single importance sample.

3 ALTERNATIVE ANNEALING PATHS FOR THE EVALUATION OF TOPIC MODELS

The AIS method described above can be very accurate if given enough computation time (Wallach et al., 2009b). However, it is subject to several potentially avoidable sources of variability. Firstly, the method estimates the ratio of the desired quantity $Pr(w^{(d)}|\Phi, \alpha)$ and the denominator $\sum_{z^{(d)}} Pr(z^{(d)}|\alpha)$ in Equation 2, which equals one, introducing stochastic noise on behalf of the denominator even though this is a constant. We would also expect that the prior may typically be very different from the posterior, thereby requiring many annealing iterations to prevent the importance weights $w^{(i)}$ from having a large variance. This has consequences for the efficiency of the sampler, which is reduced by a factor of approximately

⁴The derivation here differs slightly from that of Wallach et al. (2009b). The present derivation suggests that the procedure described in Wallach et al. produces just one importance sample. This may be repeated, finally producing as output the average of the resulting importance weights. In practice however, we found that a single sample with a longer annealing run, as in Wallach et al., may still be the best strategy on a computational budget.

$1 + \text{Var}_q[w^{(i)} / E_q[w^{(i)}]]$ relative to direct sampling from the target density (Neal, 2001).⁵

Making matters worse, we typically must perform the AIS procedure many times across all held-out documents, and therefore have a relatively limited computational budget per document, preventing us from compensating for the high variance by collecting many importance samples with a large number of temperatures. In this section, we introduce new AIS annealing paths for the evaluation of topic models which can have lower variance than the standard approach. We first introduce AIS paths which compare two topic models by annealing between them. We then show how to use these paths for evaluating topic model learning algorithms by computing per-iteration predictive performance efficiently, reusing all previous computation.

3.1 COMPARING TOPIC MODELS BY ANNEALING BETWEEN THEM

The most typical evaluation scenario is model comparison—we want to determine whether a particular model (model 1) performs better at predicting held-out documents than a baseline method (model 2) such as vanilla LDA or a model trained using a previous learning algorithm. Thus, in such situations, the quantity of interest is the *relative* log-likelihood score of the model and the baseline:

$$\begin{aligned} &\log Pr(w^{(d)}|\Phi^{(1)}, \alpha^{(1)}) - \log Pr(w^{(d)}|\Phi^{(2)}, \alpha^{(2)}) \\ &= \log \frac{Pr(w^{(d)}|\Phi^{(1)}, \alpha^{(1)})}{Pr(w^{(d)}|\Phi^{(2)}, \alpha^{(2)})}. \end{aligned} \quad (3)$$

To compute this in the framework proposed by Wallach et al., we must perform the AIS procedure once for each model, incurring the stochastic error twice. To avoid this and the aforementioned sources of variability with that approach, and given that the procedure is already designed to compute a ratio, we propose to instead use AIS to compute Equation 3 directly. Let $f_0(z^{(d)}) = Pr(w^{(d)}, z^{(d)}|\Phi^{(1)}, \alpha^{(1)})$ and $f_n(z^{(d)}) = Pr(w^{(d)}, z^{(d)}|\Phi^{(2)}, \alpha^{(2)})$. Then the desired quantity can be estimated via

$$\begin{aligned} \sum \frac{w^{(i)}}{N} &\approx \frac{\sum_{z^{(d)}} Pr(w^{(d)}, z^{(d)}|\Phi^{(1)}, \alpha^{(1)})}{\sum_{z^{(d)}} Pr(w^{(d)}, z^{(d)}|\Phi^{(2)}, \alpha^{(2)})} \\ &= \frac{Pr(w^{(d)}|\Phi^{(1)}, \alpha^{(1)})}{Pr(w^{(d)}|\Phi^{(2)}, \alpha^{(2)})}. \end{aligned} \quad (4)$$

We will refer to this strategy as “ratio-AIS.” To implement this method, it remains to choose the annealing path, i.e. the sequence of intermediate distributions. We first consider a geometric average $f_j(z^{(d)}) =$

⁵Note that $E_q[w^{(i)}]$ is equal to the ratio of normalizing constants of the target and proposal densities, which in our case is the quantity of interest, e.g. the likelihood.

$f_0(z^{(d)})^{\beta_j} f_n(z^{(d)})^{1-\beta_j}$ of the initial and final distributions, a strategy suggested by Neal (2001) with analogy to simulated annealing, where β_j can be viewed as an “inverse temperature.” To choose a transition operator T_j invariant to f_j , we straightforwardly select the Gibbs sampler. We have importance weights

$$\begin{aligned} w^{(i)} &= \prod_{j=0}^{n-1} \frac{Pr(w^{(d)}, z_j^{(d)} | \Phi^{(1)}, \alpha^{(1)})^{\beta_j}}{Pr(w^{(d)}, z_j^{(d)} | \Phi^{(1)}, \alpha^{(1)})^{\beta_{j+1}}} \\ &\times \prod_{j=0}^{n-1} \frac{Pr(w^{(d)}, z_j^{(d)} | \Phi^{(2)}, \alpha^{(2)})^{1-\beta_j}}{Pr(w^{(d)}, z_j^{(d)} | \Phi^{(2)}, \alpha^{(2)})^{1-\beta_{j+1}}} \\ &= \prod_{j=0}^{n-1} \frac{Pr(w^{(d)}, z_j^{(d)} | \Phi^{(1)}, \alpha^{(1)})^\tau}{Pr(w^{(d)}, z_j^{(d)} | \Phi^{(2)}, \alpha^{(2)})^\tau} \\ \log w^{(i)} &= \frac{1}{n} \sum_{j=0}^{n-1} \log \frac{Pr(w^{(d)}, z_j^{(d)} | \Phi^{(1)}, \alpha^{(1)})}{Pr(w^{(d)}, z_j^{(d)} | \Phi^{(2)}, \alpha^{(2)})}, \quad (5) \end{aligned}$$

assuming $\beta_j - \beta_{j+1} = \tau = n^{-1} \forall j, 0 \leq j < n-1$. Elegantly, the log importance weights are the average of the log ratios of the probabilities of $w^{(d)}$ and $z^{(d)}$ according to each model. Observe that the same z assignments are used for the numerator and denominator in each of the ratios in Equation 5, further reducing the variance of the estimate relative to the standard AIS strategy.

Although geometric averages are the standard choice for an annealing path, in many cases there exist annealing paths which perform much better. Grosse et al. (2013) introduced an alternative annealing path for exponential families which converges much more quickly, constructed by annealing averages of the moments of the sufficient statistics. The Dirichlet-multinomial distribution $Pr(z^{(d)} | \alpha)$ is not an exponential family so their method does not directly apply to LDA. Nevertheless, we consider an annealing path inspired by their work, where intermediate distributions are constructed by taking convex combinations of the parameters:

$$\begin{aligned} f_j(z^{(d)}) &= Pr(w^{(d)}, z^{(d)} | \Phi_j = \beta_j \Phi^{(1)} + (1 - \beta_j) \Phi^{(2)}, \\ \alpha_j &= \beta_j \alpha^{(1)} + (1 - \beta_j) \alpha^{(2)}). \quad (6) \end{aligned}$$

The intermediate distributions are topic models, so we set T_j to be the corresponding Gibbs sampler. This T_j does not require power operations, providing substantial execution time savings over the geometric path and Equation 2. The importance weights are

$$\begin{aligned} \log w^{(i)} &= \sum_{j=0}^{n-1} \left(\log Pr(w^{(d)}, z_j^{(d)} | \Phi_j, \alpha_j) \right. \\ &\quad \left. - \log Pr(w^{(d)}, z_j^{(d)} | \Phi_{j+1}, \alpha_{j+1}) \right). \quad (7) \end{aligned}$$

To implement this method we need to draw initially from $f_n(z^{(d)})$, which we accomplish via Gibbs sampling. These

initial samples from $f_n(z^{(d)})$ need not be independent for the procedure to work, although we may choose to run independent chains if the cost of burn-in is deemed to be less than the time wasted due to running the annealing on correlated samples. Finally, AIS will be more likely to converge if the initial and target distributions are similar to each other. We therefore align the topics before running the algorithm, using the Hungarian algorithm to minimize the L1 distances between topics. This operation, which is $O(K^3)$, where K is the number of topics, is not a computational bottleneck (relative to performing AIS) and needs only to be performed once per corpus. Pseudo-code for ratio-AIS using the path from Equation 6 is given in Algorithm 1.

Algorithm 1 Ratio-AIS, using the convex path

```

for  $i = 1 : S$  //importance samples
   $\log \omega[i] := 0$ 
   $\Phi^{(next)} := \Phi^{(2)}$ 
   $\alpha^{(next)} := \alpha^{(2)}$ 
  draw  $z^{(i)} \sim Pr(z | \alpha^{(2)})$ 
  for  $j = n-1, n-2, \dots, 0$  //temperatures
     $\Phi^{(curr)} := \Phi^{(next)}$ 
     $\alpha^{(curr)} := \alpha^{(next)}$ 
     $\Phi^{(next)} := \beta_j \Phi^{(1)} + (1 - \beta_j) \Phi^{(2)}$ 
     $\alpha^{(next)} := \beta_j \alpha^{(1)} + (1 - \beta_j) \alpha^{(2)}$ 
    for  $a = 1 : r_j$  //  $r_{n-1}$  is large, for burn in
      for  $l = 1 : \text{length}(w^{(d)})$  //words
        //draw  $z_l^{(i)}$ 
         $Pr(z_l^{(i)} = k | \cdot) \propto (n_k^{(i)} + \alpha_k^{(curr)}) \Phi_{w_l^{(d)}, k}^{(curr)}$ 
       $\log \omega[i] := \log \omega[i]$ 
       $+ \log Pr(w^{(d)}, z^{(i)} | \Phi^{(next)}, \alpha^{(next)})$ 
       $- \log Pr(w^{(d)}, z^{(i)} | \Phi^{(curr)}, \alpha^{(curr)})$ 
  return  $\log \text{SumExp}(\log \omega) - \log(S)$ 

```

Detecting Convergence Failures

AIS can produce poor estimates if the annealing fails to converge to a high-probability state in the target distribution within the given set of iterations. In general, this may be very difficult to detect. However, in our case we can interchange f_0 and f_n in our AIS strategy to compute the reciprocal of the desired ratio, and compare the reciprocal of this to our estimate. If these two values are wildly different, then we will know that the annealing has failed to converge. This means that we are able to detect convergence failures in many practical cases. In our experiments, we were easily able to catch convergence failures by observing a systematic bias across documents in the results of the different annealing directions (see Section 4).

3.2 EFFICIENTLY EVALUATING TOPIC MODEL LEARNING ALGORITHMS WITH ITERATION-AIS

When evaluating algorithms for learning topic models (or monitoring their convergence), we would ideally like to compute and plot held-out log-likelihood scores per learning iteration (or unit of computation time) for each algorithm under consideration. This is extremely expensive, requiring $|H| \times I \times M$ Monte Carlo approximations of already intractable high-dimensional integrals, where H is the held-out test set, I is the number of iterations of the learning algorithms to evaluate at, and M is the number of competing learning methods.

Fortunately, for many learning algorithms such as the collapsed Gibbs sampler, the topics at successive iterations are similar to each other, and the topics typically vary smoothly from “high temperature” high entropy distributions at early iterations to more complicated later distributions. This suggests using a single AIS path to perform the entire evaluation across all of the iterations, with the topic models at each iteration (or a subset of them) as intermediate distributions. We can accomplish this by annealing from the prior $Pr(z^{(d)}|\alpha^{(1)})$ to the first topic model $Pr(w^{(d)}, z^{(d)}|\Phi^{(1)}, \alpha^{(1)})$ as in Wallach et al. (2009b), and then using ratio-AIS to anneal between successive topic models $Pr(w^{(d)}, z^{(d)}|\Phi^{(t)}, \alpha^{(t)})$. For the topic model at iteration t , the average $S^{-1} \sum_i w^{(i,t)}$ of the importance weights computed up to that point $w^{(i,t)}$ converges to the ratio of normalizing constants,

$$\frac{\sum_{z^{(d)}} Pr(w^{(d)}, z^{(d)}|\Phi^{(t)}, \alpha^{(t)})}{\sum_{z^{(d)}} Pr(z^{(d)}|\alpha^{(1)})} = Pr(w^{(d)}|\Phi^{(t)}, \alpha^{(t)}). \quad (8)$$

With n temperatures per learning iteration k , importance weights can be written recursively as

$$\begin{aligned} \log w^{(i,t)} &= \sum_{t'=1}^t \sum_{j=0}^{n-1} \log \frac{f_{t',j}(z_{t',j})}{f_{t',j+1}(z_{t',j})} \\ &= \log w^{(i,t-1)} + \sum_{j=0}^{n-1} \log \frac{f_{t,j}(z_{t,j})}{f_{t,j+1}(z_{t,j})}. \end{aligned} \quad (9)$$

This method, which we refer to as *iteration-AIS*, exploits all of the computation for selecting z assignments and importance weights from the likelihood estimates at previous learning iterations, leading to successively longer annealing runs, and therefore potentially better likelihood estimates, as k increases.

4 EXPERIMENTS

We explored the performance of the proposed techniques using a corpora of scientific articles from the Association

of Computational Linguistics (ACL) conference⁶ (Radev et al., 2013), and another from the Neural Information Processing Systems (NIPS) conference.⁷ The ACL dataset consists of the 3286 articles from the years 1987 to 2011, while the NIPS corpus contains the 1740 articles published between 1987 and 1999. In each experiment, topic models with 50 topics were fit to each corpus by performing 1000 iterations of collapsed Gibbs sampling using the MALLET toolkit (McCallum, 2002). Roughly 10% of the documents in each corpus were withheld for testing (130 NIPS articles, and 300 ACL articles). Although cross-validation would have been a preferable option to using a single hold-out set, the computational expense of the experiments prevented this. For example, across all algorithms and learning iterations, Figure 2 required a total of 6.6 million Gibbs iterations for each one of the 300 test articles.

When using AIS we must select the number of temperatures n , the number of importance samples S , and the temperature schedule $\beta_0, \beta_1, \dots, \beta_n$. The variability of an AIS estimator can be reduced by increasing S (due to the law of large numbers) or by increasing n (which reduces the variance of the $w^{(i)}$). In the experiments, we focused on the case where $S = 1$, as in Wallach et al. (2009b). We found in preliminary experiments that $S = 1$ gave essentially exactly the same answer as $S = 100$ importance samples for Ratio-AIS with 10,000 temperatures. For simplicity, we used a uniform spacing of the temperatures β_j .⁸

We also compared to the left-to-right (LR) particle filtering algorithm of Wallach et al. (2009b), using the implementation provided in MALLET. The left-to-right method requires $N_d(N_d + 1)/2$ word-level Gibbs updates per particle for a document of length N_d . The execution of $p = 2 \times n/(N_d + 1)$ particles corresponds to the same number of Gibbs updates as AIS with n temperatures and $S = 1$. We select the number of LR particles by rounding p to the nearest integer greater than zero.

Ratio-AIS was designed for reliable per-document comparisons. To explore this, we ran each algorithm twice on each document, and reported results comparing the two runs across documents. To remove the effect of document length in the results, instead of reporting the differences in log-likelihood scores for each model we consider instead perplexity scores $\exp(-\frac{\log Pr(w^{(d)}|\Phi, \alpha)}{N_d})$. The ratio of the perplexity of model 1 over the perplexity of model 2 for a document is easily computed from the output of Ratio-AIS as $\exp(\frac{L_2 - L_1}{N_d})$, where L_j is the log-likelihood for model j . We considered two evaluation scenarios: comparing learned topics to perturbed versions of the same top-

⁶Available at <http://clair.eecs.umich.edu/aan/index.php>.

⁷The NIPS dataset, due to Gregor Heinrich, is available at <http://www.arbylon.net/resources.html>.

⁸Neal (2001) suggests that a geometric spacing of the β_j 's may be beneficial, at least for the geometric annealing path.

ics (Section 4.1), and comparing topic models learned with symmetric and asymmetric Dirichlet priors (Section 4.2). Finally, we evaluated iteration-AIS for estimation of per-iteration likelihood (Section 4.3).

4.1 LEARNED TOPICS VERSUS PERTURBED TOPICS

As the likelihoods we are trying to estimate are intractable, we do not in general have access to ground truth. However, after learning topics Φ on a dataset and then creating a noisy copy of them Φ' , we have good reason to believe that the original topics Φ are better than the copy. This style of experiment was previously performed by Wallach et al. (2009b). We took the word-topic assignments learned by MALLET, and created Φ' by re-assigning 5% of them to new word-topic assignments uniformly at random.⁹

Ratios of the perplexities for the two models were computed with both cheap (100 temperatures) and expensive runs (10,000 temperatures). Overall results are given in Table 1, and per-document results for the ACL dataset are plotted in Figure 1.¹⁰ The two ratio-AIS paths were both the most accurate and the most consistent methods, in both temperature regimes.

In the cheap regime, the ratio-AIS points are slightly off-diagonal in Figure 1, with one annealing direction giving systematically lower results, representing a detectable bias due to convergence failure in at least one annealing direction. Nevertheless, these results have much lower variance, and the bias disappears in the expensive regime. Surprisingly, the standard AIS method performed extremely poorly, with most data points falling outside of the boundaries of the figure, which are tight around the results of the other methods. Using many importance samples would very likely mitigate this, at greater computational cost. It should be noted that the task of comparing two very similar topic models is difficult for standard methods, but is relatively easy for ratio-AIS due to the distance to anneal between the distributions being smaller.

4.2 SYMMETRIC VERSUS ASYMMETRIC DIRICHLET PRIORS

Learning asymmetric α hyperparameters can improve the predictive performance of topic models (e.g., Wallach et al. (2009a)). To explore this, on each corpus we learned a topic model with asymmetric α , and a model where α was fixed to be flat but its concentration parameter was learned. The AIS and LR algorithms were used to compare the resulting models, using runs with 1000 temperatures and 10,000 temperatures.

⁹MALLET’s left-to-right takes as input a count matrix, so the perturbed topics must be representable as counts.

¹⁰Results for the NIPS corpus are similar, and are provided in Foulds (2014).

It was found that in the “cheap” 1000 temperature regime, the ratio-AIS estimates were the most closely correlated with left-to-right estimates in the expensive regime, the best available proxy for ground truth (Table 2, top).¹¹ In all cases the ratio-AIS paths had one to two orders of magnitude lower empirical variances in the estimates of per-document perplexity ratios than the previous methods, with the convex path having the least variance (Table 2, middle). Ratio-AIS therefore achieves the original goal of greatly reducing the variance of per-document comparisons of topic models. This is particularly important if we want to perform detailed analysis at a per-document level, such as exploring the effect of covariates on topic model performance. In such a scenario, the previous methods have unacceptably high variance for a reasonable level of computation (see also Figure 1), while the ratio-AIS estimates of relative performance have very small empirical variance with just one importance sample.

Unfortunately, this reduction comes at a price of potentially increased bias in the estimated perplexity ratio when given insufficient computation. Topic models which learn an asymmetric α tend to perform better than those with a symmetric α (Wallach et al., 2009a), and the previous methods detected a larger advantage for the asymmetric approach (Table 2, bottom). The direction of the ratio-AIS annealing path also made a difference to the outcome. In particular, the forward direction of annealing did not detect an overall advantage to the asymmetric hyper-parameter model. On the other hand, the difference per direction allowed us to detect a convergence failure, which is difficult to do in general. Also note that for the task in Section 4.1, the overall perplexity ratios were very consistent between annealing directions, and showed a clearer difference between models than the baseline algorithms did – see Foulds (2014).

4.3 EVALUATING TOPIC MODELS PER ITERATION

The iteration-AIS annealing path evaluates the performance of topic model learning algorithms on a per-iteration basis. We explored its performance using the convex path with 1000 and 10,000 temperatures per learned model, annealing between the models at every 10th learning iteration. At the first learning iteration $\Phi^{(1)}$, the algorithms were given an extra 1000 temperatures to compensate for the cold-start from the prior.

Results on ACL are shown in Figure 2. It was found that iteration-AIS estimated higher log-likelihoods than left-to-right and standard AIS in both temperature regimes (Figure 2, left). The main failure mode of these algorithms is to underestimate the likelihood by failing to find high probability regions, so higher values are likely to be better

¹¹The standard AIS estimate of the perplexity ratios had too high a variance to be used (see Table 2).

% Correct	Left to Right	Standard AIS	Ratio-AIS Geometric	Ratio-AIS Geom. (reverse)	Ratio-AIS Convex	Ratio-AIS Convex (reverse)
NIPS (cheap)	63.8	48.8	83.8	89.2	84.6	87.7
NIPS (expensive)	84.6	62.3	86.9	87.7	87.7	87.7
ACL (cheap)	80.2	50.8	88.3	92.0	88.3	92.3
ACL (expensive)	90.7	75.2	90.3	90.3	90.3	90.3

Table 1: Percentage of documents where the learned topics Φ were estimated to have higher likelihood than the perturbed topics Φ' .

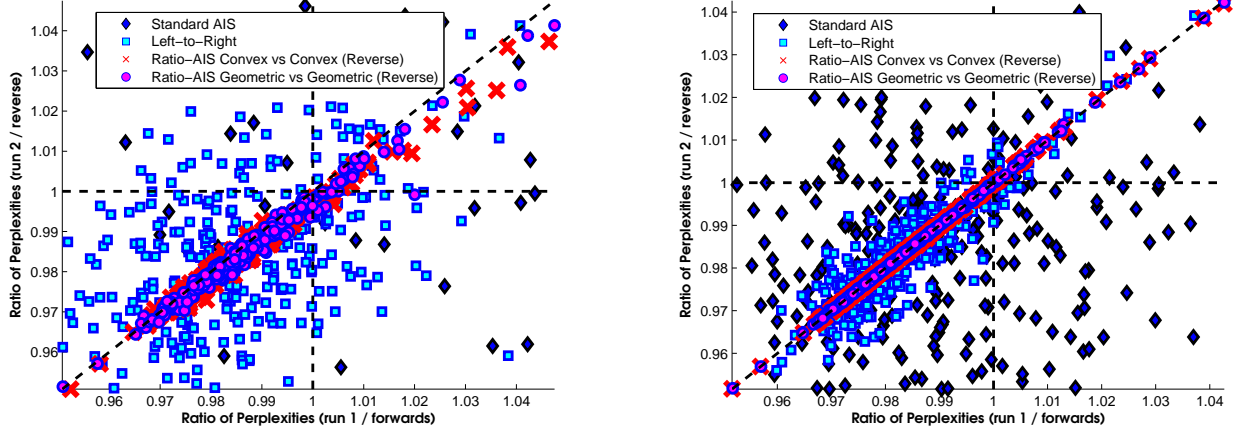


Figure 1: Comparing learned topics with perturbed versions of them, on the ACL dataset. In the figures, every point corresponds to a document. Each axis corresponds to estimated $\frac{\text{perp}(\Phi)}{\text{perp}(\Phi')}$ for a repeat of the experiment, with the ratio-AIS repeats being performed in different annealing directions. Points in the lower left quadrant are those which (likely correctly) predict the unperturbed topics as the winner in both trials. Points near the diagonal have consistent results across the two trials. **Left**: 100 temperatures. **Right**: 10,000 temperatures. Missing Standard AIS results are outside of the bounds of the plots. Figure best viewed in color.

Correlation with Long LR Run	Left to Right	Standard AIS	Ratio-AIS Geometric	Ratio-AIS Geom. (reverse)	Ratio-AIS Convex	Ratio-AIS Convex (reverse)
NIPS (cheap)	0.947	0.619	0.973	0.975	0.976	0.981
NIPS (expensive)	0.993	0.852	0.981	0.982	0.981	0.982
ACL (cheap)	0.965	0.578	0.984	0.983	0.987	0.986
ACL (expensive)	0.995	0.892	0.989	0.989	0.990	0.989

Variance of Perplexity Ratio	Left to Right	Standard AIS	Ratio-AIS Geometric	Ratio-AIS Geom. (reverse)	Ratio-AIS Convex	Ratio-AIS Convex (reverse)
NIPS (cheap)	2.6×10^{-4}	2.6×10^{-3}	2.0×10^{-5}	1.5×10^{-5}	8.2×10^{-6}	9.8×10^{-6}
NIPS (expensive)	1.7×10^{-5}	6.0×10^{-4}	1.4×10^{-6}	1.2×10^{-6}	6.9×10^{-7}	5.8×10^{-7}
ACL (cheap)	1.7×10^{-4}	3.6×10^{-3}	1.6×10^{-5}	1.3×10^{-5}	7.7×10^{-6}	6.6×10^{-6}
ACL (expensive)	1.4×10^{-5}	5.6×10^{-4}	1.1×10^{-6}	9.4×10^{-7}	7.4×10^{-7}	5.1×10^{-7}

Corpus-Level Perplexity Ratio	Left to Right	Standard AIS	Ratio-AIS Geometric	Ratio-AIS Geom. (reverse)	Ratio-AIS Convex	Ratio-AIS Convex (reverse)
NIPS (cheap)	0.984	0.975	1.01	0.992	1.01	0.994
NIPS (expensive)	0.989	0.990	1.00	0.999	1.00	0.998
ACL (cheap)	0.984	0.980	1.00	0.985	1.00	0.988
ACL (expensive)	0.987	0.989	0.994	0.992	0.996	0.992

Table 2: Comparing asymmetric α and symmetric α topic models. Correlation coefficient with the perplexity ratio estimates from a run of left-to-right in the expensive regime (**top**), average empirical variance (evaluated across two runs per document) of the per-document perplexity ratio (**middle**), and the overall perplexity ratio for the entire corpus (**bottom**).

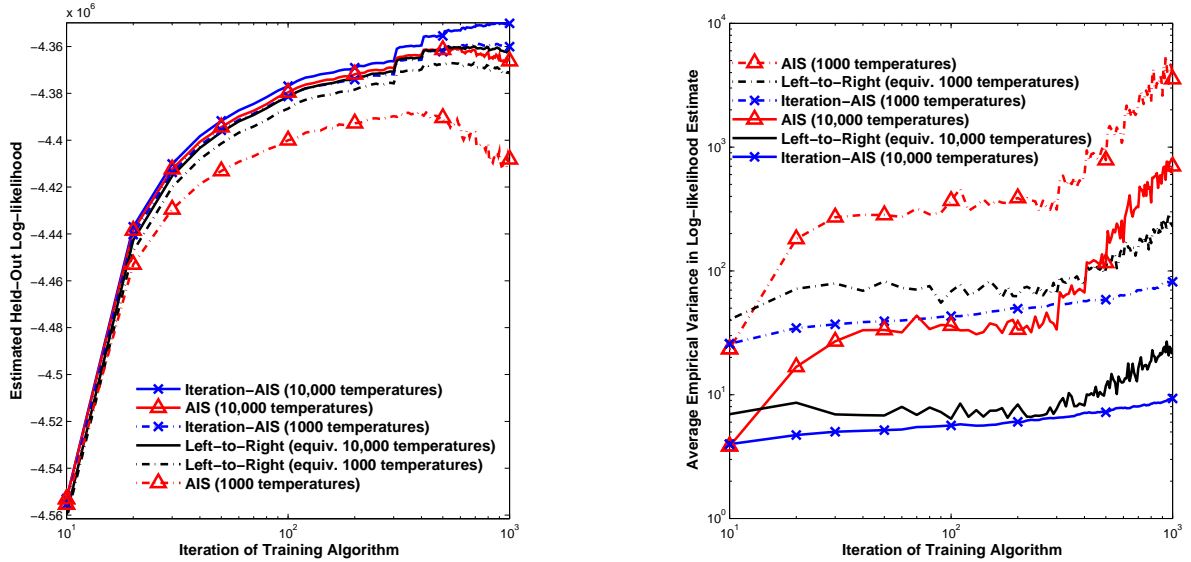


Figure 2: Evaluating iteration-AIS on ACL. Jumps in log-likelihood are due to hyper-parameter optimization. Figure best viewed in color.

(Wallach et al., 2009b). Consistent with this observation, the iteration-AIS likelihood curve at 1000 temperatures coincided with the likelihood curves of the baselines when they were given ten times more computation. The proposed method also exhibited much lower variance in the likelihood estimates (Figure 2, right). This is expected, as the effective number of annealing temperatures is higher, which is known to reduce the variance of the importance weights (Neal, 2001). Similar results were observed on NIPS (see Foulds (2014) for these and other additional results).

The baselines reported decreasing held-out likelihood in later iterations, while iteration-AIS did not. Such a decrease could be due to over-fitting, but is more likely to be caused by convergence failures due to the topics becoming more complex. As evidence for this, the dip in likelihood was smaller with increased computation, and all methods exhibited higher variance in the likelihood estimates for later learning iterations (Figure 2, right, computed based on two evaluations of the likelihood per document, and averaged across documents). The prior probability of the topic models also decreased from around iteration 300 (the same point where standard AIS began to report a decrease in performance), and this is likely to make inference more difficult (see Foulds (2014)).

5 CONCLUSIONS

We have introduced ratio-AIS, a strategy for comparing topic models, and empirically evaluated its performance relative to previous methods using two datasets. Ratio-AIS was found to have low empirical variance, making it useful for document-level analysis. It should be noted that importance sampling can suffer from bias with a finite number of

samples, e.g. approaches such as those described by Wallach et al. (2009b) will typically underestimate the likelihood. For ratio-AIS this results in the potential for a bias that favors a particular model when an insufficient number of samples or temperatures is used, due to the directional nature of the approach. Such a convergence failure of a Monte Carlo algorithm is in general very difficult to detect, but in the proposed method the bias is frequently easily detectable by comparing the results of two Monte Carlo runs. When applied to the evaluation of the per-iteration performance of topic model training algorithms (iteration-AIS), the method outperforms traditional approaches even when given an order of magnitude less computation. Based on our results, we recommend ratio-AIS for document-level analysis, or in cases where the topics are very similar to each other. The method should be performed using both annealing directions as a convergence sanity check, at least for a subset of the held-out documents. Left-to-right is still generally preferred for corpus-level perplexity comparisons, unless per-iteration curves are desired, in which case we recommend that iteration-AIS be used.

For future work, it is straightforward to adapt ratio-AIS to the document completion task. It may also be possible to find other AIS paths with better mixing properties, and the ideas in this work may be applicable to other latent variable models such as RBMs. See Foulds (2014) for a discussion on these ideas, as well as on the use of ratio-AIS with multiple topic models, and where the models differ in the number of topics or in their parametric forms.

Acknowledgements

This work was supported by the Office of Naval Research under MURI grant N00014-08-1-1015.

References

- Blei, D.M., Ng, A.Y., and Jordan, M.I. Latent Dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022, 2003.
- Buntine, W. Estimating likelihoods for topic models. In *Advances in Machine Learning. First Asian Conference on Machine Learning, ACML 2009, Nanjing, China, November 2-4, 2009. Proceedings*, volume 5828 of *Lecture Notes in Computer Science*, pp. 51–64. Springer, 2009.
- Chang, Jonathan, Boyd-Graber, Jordan, Gerrish, Sean, Wang, Chong, and Blei, David. Reading tea leaves: How humans interpret topic models. In *Advances in Neural Information Processing Systems 22*, pp. 288–296, 2009.
- Foulds, J. R. *Latent Variable Modeling for Networks and Text: Algorithms, Models and Evaluation Techniques*. PhD thesis, University of California, Irvine, 2014.
- Foulds, J. R., Boyles, L., DuBois, C., Smyth, P., and Welling, M. Stochastic collapsed variational Bayesian inference for latent Dirichlet allocation. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 446–454, 2013.
- Grimmer, Justin. A Bayesian hierarchical topic model for political texts: Measuring expressed agendas in Senate press releases. *Political Analysis*, 18(1):1–35, 2010.
- Grosse, Roger, Maddison, Chris, and Salakhutdinov, Ruslan. Annealing between distributions by averaging moments. In *Advances in Neural Information Processing Systems 26*, pp. 2769–2777, 2013.
- Hoffman, Matt, Blei, David M, Wang, Chong, and Paisley, John. Stochastic variational inference. *Journal of Machine Learning Research*, 14(1):1303–1347, 2013.
- Hoffman, Matthew, Bach, Francis R, and Blei, David M. Online learning for latent Dirichlet allocation. In *Advances in Neural Information Processing Systems*, pp. 856–864, 2010.
- McCallum, Andrew Kachites. Mallet: A machine learning for language toolkit. <http://mallet.cs.umass.edu>, 2002.
- McFarland, Daniel A, Ramage, Daniel, Chuang, Jason, Heer, Jeffrey, Manning, Christopher D, and Jurafsky, Daniel. Differentiating language usage through topic models. *Poetics*, 41(6):607–625, 2013.
- Mimno, D. and McCallum, A. Topic models conditioned on arbitrary features with Dirichlet-multinomial regression. In *Proceedings of the Twenty-Fourth International Conference on Uncertainty in Artificial Intelligence*, pp. 411–418, 2008.
- Mimno, David, Wallach, Hanna M, Talley, Edmund, Leenders, Miriam, and McCallum, Andrew. Optimizing semantic coherence in topic models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 262–272. Association for Computational Linguistics, 2011.
- Neal, R.M. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, 2001.
- Newman, David, Lau, Jey Han, Grieser, Karl, and Baldwin, Timothy. Automatic evaluation of topic coherence. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 100–108. Association for Computational Linguistics, 2010.
- Nguyen, Viet-An, Boyd-Graber, Jordan, Resnik, Philip, Cai, Deborah A, Midberry, Jennifer E, and Wang, Yuanxin. Modeling topic control to detect influence in conversations using nonparametric topic models. *Machine Learning*, pp. 1–41, 2013.
- Radev, Dragomir R., Muthukrishnan, Pradeep, Qazvinian, Vahed, and Abu-Jbara, Amjad. The ACL anthology network corpus. *Language Resources and Evaluation*, 47(4):919–944, 2013.
- Rosen-Zvi, M., Griffiths, T., Steyvers, M., and Smyth, P. The author-topic model for authors and documents. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pp. 487–494. AUAI Press, 2004.
- Scott, J.G. and Baldridge, J. A recursive estimate for the predictive likelihood in a topic model. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, pp. 527–535, 2013.
- Wallach, Hanna M, Mimno, David M, and McCallum, Andrew. Rethinking LDA: Why priors matter. In *Advances in Neural Information Processing Systems 22*, pp. 1973–1981, 2009a.
- Wallach, H.M., Murray, I., Salakhutdinov, R., and Mimno, D. Evaluation methods for topic models. In *International Conference on Machine Learning*, pp. 1105–1112. ACM, 2009b.

Active Learning of Linear Embeddings for Gaussian Processes

Roman Garnett

University of Bonn
Römerstraße 164
53117 Bonn, Germany
rgarnett@uni-bonn.de

Michael A. Osborne

University of Oxford
Parks Road
Oxford OX1 3PJ, UK
mosb@robots.ox.ac.uk

Philipp Hennig

MPI for Intelligent Systems
Spemannstraße
72076 Tübingen, Germany
phennig@tue.mpg.de

Abstract

We propose an active learning method for discovering low-dimensional structure in high-dimensional Gaussian process (GP) tasks. Such problems are increasingly frequent and important, but have hitherto presented severe practical difficulties. We further introduce a novel technique for approximately marginalizing GP hyperparameters, yielding marginal predictions robust to hyperparameter misspecification. Our method offers an efficient means of performing GP regression, quadrature, or Bayesian optimization in high-dimensional spaces.

1 INTRODUCTION

We propose a method to actively learn, simultaneously, about a function and a low-dimensional embedding of its input domain. High dimensionality has stymied the progress of model-based approaches to many common machine learning tasks. In particular, although Bayesian nonparametric modeling with Gaussian processes (GPs) (Rasmussen & Williams, 2006) has become popular for regression, classification, quadrature (O’Hagan, 1991), and global optimization (Brochu et al., 2010), such approaches remain intractable for large numbers of input variables (with the exception of local optimization (Hennig & Kiefel, 2012)). An old idea for the solution to this problem is the exploitation of low-dimensional structure; the most tractable such case is that of a linear embedding. Throughout this text, we consider a function $f(x): \mathbb{R}^D \rightarrow \mathbb{R}$ of a high-dimensional variable $x \in \mathbb{R}^D$ (for notational simplicity, x will be assumed to be a row vector). The assumption is that f , in reality, only depends on the variable $u := xR^\top$, of much lower dimensionality $d \ll D$, through a linear embedding $R \in \mathbb{R}^{d \times D}$. We are interested in an algorithm that simultaneously learns R and f , and does so in an active way. That is, it iteratively selects informative locations x_* in a box-bounded region $\mathcal{X} \subset \mathbb{R}^D$, and collects associated obser-

vations y_* of $f_* := f(x_*)$ corrupted by i.i.d. Gaussian noise: $p(y_* | f_*) = \mathcal{N}(y_*; f_*, \sigma^2)$.

The proposed method comprises three distinct steps (Algorithm 1): constructing a probability distribution over possible embeddings (*learning the embedding R*); using this belief to determine a probability distribution over the function itself (*learning the function f*), and then choosing evaluation points to best inform these beliefs (*active selection*). To learn the embedding, we use a Laplace approximation on the posterior over R to quantify the uncertainty in the embedding (Section 2). To learn the function, we develop a novel approximate means of marginalizing over Gaussian process hyperparameters (including those parameterizing embeddings), to provide predictions robust to hyperparameter misspecification (Section 3). This sub-algorithm is more generally applicable to many Gaussian process tasks, and to the marginalization of hyperparameters other than embeddings, and so represents a core contribution of this paper. Finally, for active selection, we extend previous work (Houlsby et al., 2011) to select evaluations that maximize the expected reduction in uncertainty about R (Section 4).

Estimators for R in wide use include LASSO (Tibshirani, 1996) and the Dantzig selector (Candes & Tao, 2007), both of which assume $d = 1$. These are passive methods estimating the linear embedding from a fixed dataset. This paper develops an algorithm that *actively* learns R for the domain of a Gaussian process. The goal is to use few function evaluations to intelligently explore and identify R . Notice that although the embedding is assumed to be linear, the function f itself will be allowed to be nonlinear via the GP prior.

This problem is related to, but distinct from, dimensionality reduction (Lawrence, 2012), for which active learning has recently been proposed (Iwata et al., 2013). Dimensionality reduction is also known as visualization or blind source separation, and is solved using, e.g., principal component analysis (PCA), factor analysis, or latent variable models. As in dimensionality reduction, we consider the problem of finding a low-dimensional representation of an input or feature matrix $X \in \mathbb{R}^{N \times D}$; unlike dimensionality reduction, we do so given an associated vector of training

Algorithm 1 Simultaneous active learning of functions and their linear embeddings (pseudocode)

Require: d, D ; kernel κ , mean function μ ; prior $p(R)$

$X \leftarrow \emptyset; y \leftarrow \emptyset$

repeat

$q(R) \leftarrow \text{LAPLACEAPPROX}(p(R | X, y, \kappa, \mu))$
 // approximate posterior on embedding R

$q(f) \leftarrow \text{APPROXMARGINAL}(p(f | R), q(R))$
 // approximate marginal on function f

$x_* \leftarrow \text{OPTIMIZEUTILITY}(q(f), q(R))$
 // find approximate optimal evaluation point x_*

$y_* \leftarrow \text{OBSERVE}(f(x_*))$ // act

$X \leftarrow [X; x_*]; y \leftarrow [y; y_*]$ // store data

until budget depleted

return $q(R), q(f)$.

outputs or labels $y \in \mathbb{R}^N$, containing information about which inputs are most relevant to a function. The problem of discovering linear embeddings of GPs was discussed by Snelson & Ghahramani (2006) for the passive case. Active supervised learning has been widely investigated (MacKay, 1992b; Guestrin et al., 2005; Houlby et al., 2011); our work hierarchically extends this idea to additionally identify the embedding. A special case of our method (the case of a diagonal R) is the hitherto unconsidered problem of *active* automatic relevance determination (MacKay, 1992a; Neal, 1995; Williams & Rasmussen, 1996).

Identifying embeddings is relevant for numerous Gaussian process applications, notably regression, classification, and optimization. Within Bayesian optimization, much recent work has focused on high-dimensional problems (Hutter et al., 2011; Chen et al., 2012; Carpentier & Munos, 2012; Bergstra & Bengio, 2012; Hutter, 2009). Recently, Wang et al. (2013) proposed using randomly generated linear embeddings. In contrast, our active learning strategy can provide an initialization phase that selects objective function evaluations so as to best learn low-dimensional structure. This permits the subsequent optimization of high-dimensional objectives over only the learned low-dimensional embedding.

Alongside this paper, we are releasing open-source software implementing our contributions. A simple MATLAB implementation of Algorithm 1, built on the Gaussian Process for Machine Learning (GPML) Toolbox,¹ is freely available.² An independent, GPML-compatible implementation of the MGP (Section 3) is also available.³

¹<http://www.gaussianprocess.org/gpml/code>

²https://github.com/rmgarnett/active_gp_hyperlearning

³<https://github.com/rmgarnett/mgp>

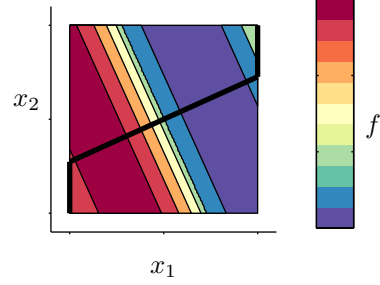


Figure 1: A function f with a one-dimensional linear embedding and its box-bounded domain \mathcal{X} . Searching over the thick black lines captures all variation of the function. Our aim is to learn this embedding, represented by embedding matrix $R \in \mathbb{R}^{1 \times 2}$, by selecting evaluations of f in some search space.

2 LINEAR EMBEDDINGS OF GAUSSIAN PROCESSES

In many applications, like image analysis, D can be in the order of thousands or millions. But even $D = 10$ is a high dimensionality for common Gaussian process models, not only from a computational, but also from an informational, perspective. Because standard GP covariance functions stipulate that function values separated by more than a few input length scales are negligibly correlated, for high D , almost all of \mathcal{X} is uncorrelated with observed data. Hence data is effectively ignored during most predictions, and learning is impossible. Practical experience shows, however, that many functions are insensitive to some of their inputs (Wang et al., 2013), thus have low *effective* dimensionality (Figure 1). Our goal is to discover an $R \in \mathbb{R}^{d \times D}$ such that, for low-dimensional $\mathcal{U} \subset \mathbb{R}^d$, $u = xR^\top$, $\forall u \in \mathcal{U}$, $x \in \mathcal{X}$ and $\tilde{f}(x) = \tilde{f}(u)$ for a new function, $\tilde{f}: \mathcal{U} \rightarrow \mathbb{R}$. The discussion here will be restricted to predefined d ; in reality, this is likely to be defined as the maximum number of dimensions that can be feasibly considered in light of computational constraints. If the actual d is lower than this limit, R can be padded with rows of zeros.

We adopt a GP prior on \tilde{f} with mean and covariance functions $\tilde{\mu}$ and $\tilde{\kappa}$, respectively. The linear embedding induces another GP prior $p(f) = \mathcal{GP}(f; \mu, \kappa)$, where $\mu(x) = \tilde{\mu}(xR^\top)$ and $\kappa(x, x') = \tilde{\kappa}(xR^\top, x'R^\top)$. For example, if $\tilde{\kappa}$ is the well-known isotropic exponentiated-quadratic (squared-exponential, radial basis function, Gaussian) covariance, $\tilde{\kappa}(u, u') := \gamma^2 \exp[-\frac{1}{2}(u - u')(u - u')^\top]$ with output scale γ , κ on f is the Mahalanobis exponentiated-quadratic covariance

$$\kappa(x, x') = \gamma^2 \exp\left[-\frac{1}{2}(x - x')R^\top R(x - x')^\top\right]. \quad (1)$$

If $d = D = 1$, then $R \in \mathbb{R}$ is an inverse length scale. We will return to this one-dimensional example later to build intuition. A further special case is a diagonal R (assuming

$d = D$), in which case κ is the automatic relevance determination (ARD) covariance (Neal, 1995), widely used to identify the most-important inputs.

Given an appropriate R with acceptably small d , learning about f is possible even for large D , because the regression problem is reduced to the manageable space \mathbb{R}^d . This can remain true even in the case of an uncertain R : in particular, assume a prior $p(R) = \mathcal{N}(R; \hat{R}, \Sigma)$. Thus, recalling that $u = xR^\top$, and using standard Gaussian identities, if $d = 1$, $p(u | x) = \mathcal{N}(u; x\hat{R}^\top, x\Sigma x^\top)$. If $d > 1$, Σ is $\text{Cov}[\text{vect}R]$, resulting in another Gaussian for $p(u | x)$ that is only slightly more involved than in the $d = 1$ case. As such, regression on f reduces to GP regression on \tilde{f} , whose domain is the much smaller $\mathcal{U} \subset \mathbb{R}^d$, but with uncertain, Gaussian-distributed, inputs. Unlike the work of McHutchon & Rasmussen (2011), giving an existing approach to GP regression with uncertain inputs, the Gaussian over the inputs here is correlated; the location of a point is correlated with all others via mutual dependence on R . And unlike the setting considered by Girard & Murray-Smith (2005), there is no natural ordering of this domain enabling an iterative procedure. The following section describes a novel means of regression with uncertain embedding R .

2.1 APPROXIMATING THE POSTERIOR ON R

The log-likelihood of R , after N observations forming a dataset $\mathcal{D} := (X, y) \in \mathbb{R}^{N \times D} \times \mathbb{R}^N$, is

$$\begin{aligned} \log p(y | X, R) &= \log \mathcal{N}(y; \mu_X, K_{XX} + \sigma^2 \mathbb{I}) \\ &= -1/2[(y - \mu_X)^\top (K_{XX} + \sigma^2 \mathbb{I})^{-1} (y - \mu_X) \\ &\quad + \log |K_{XX} + \sigma^2 \mathbb{I}| + N \log 2\pi]. \end{aligned} \quad (2)$$

As $\mu_X := \mu(X)$ and $K_{XX} := \kappa(X, X)$ have nonlinear dependence upon R , so does $p(y | X, R)$. Even a simplistic prior on the elements of R thus gives a complicated posterior. We will use a Laplace approximation for $p(R | \mathcal{D})$ to attain a tractable algorithm. To construct a Gaussian approximation, $\mathcal{N}(R; \hat{R}, \Sigma) \simeq p(R | \mathcal{D})$, we find a mode of the posterior of $p(R | \mathcal{D})$ and set this mode as the mean \hat{R} of our approximate distribution. The covariance of the Gaussian approximation is taken as the inverse Hessian of the negative logarithm of the posterior evaluated at \hat{R} ,

$$\Sigma^{-1} = -\nabla \nabla^\top \log p(R | \mathcal{D}) \Big|_{R=\hat{R}}. \quad (3)$$

2.1.1 Computational Cost

How costly is it to construct the Laplace approximation of Equation (3)? Since D may be a large number, active learning should have low cost in D . This section shows that the required computations can be performed in time linear in D , using standard approximate numerical methods. It is a technical aspect that readers not interested in details may want to skip over.

Up to normalization, the log posterior is the sum of log prior and log likelihood (2). The former can be chosen very simplistically; the latter has gradient and Hessian given by, defining $G := \kappa_{XX} + \sigma^2 \mathbb{I}$ and $\Gamma := G^{-1}(y - \mu_X)$,

$$\begin{aligned} -2 \frac{\partial \log p(y | X, R)}{\partial \theta} &= -\Gamma^\top \frac{\partial \kappa_{XX}}{\partial \theta} \Gamma + \text{Tr} \left[G^{-1} \frac{\partial \kappa_{XX}}{\partial \theta} \right]; \\ -2 \frac{\partial^2 \log p(y | X, R)}{\partial \theta \partial \eta} &= 2\Gamma^\top \frac{\partial \kappa_{XX}}{\partial \eta} G^{-1} \frac{\partial \kappa_{XX}}{\partial \theta} \Gamma \\ &\quad - \text{Tr} \left[G^{-1} \frac{\partial \kappa_{XX}}{\partial \eta} G^{-1} \frac{\partial \kappa_{XX}}{\partial \theta} \right] \\ &\quad - \Gamma^\top \frac{\partial^2 \kappa_{XX}}{\partial \theta \partial \eta} \Gamma + \text{Tr} \left[G^{-1} \frac{\partial^2 \kappa_{XX}}{\partial \theta \partial \eta} \right]. \end{aligned} \quad (4)$$

Together with the analogous expressions for a prior $p(R)$, these expressions can be used to find a maximum of the posterior distribution (e.g., via a quasi-Newton method), and the Hessian matrix required for the Laplace approximation to $p(R | \mathcal{D})$. The computational cost of evaluating these expressions depends on the precise algebraic form of the kernel κ . For the exponentiated quadratic kernel of Equation (1), careful analysis shows that the storage cost for the Hessian of (2) is $\mathcal{O}(N^2 d D)$, and its structure allows its multiplication with a vector in $\mathcal{O}(N^2 d D)$. The corresponding derivations are tedious and not particularly enlightening. To give an intuition, consider the most-involved term in (4): Using the short-hand $\Delta_\ell^{ij} := x_{i\ell} - x_{j\ell}$, a straightforward derivation gives the form

$$\begin{aligned} H_{k\ell, ab}^1 &:= - \sum_{ij} \Gamma_i \frac{\partial^2 \kappa(x_i, x_j)}{\partial R_{k\ell} \partial R_{ab}} \Gamma_j \\ &= \sum_{ijop} R_{ko} \Delta_o^{ij} \Delta_\ell^{ij} \Gamma_i \kappa(x_i, x_j) \Gamma_j R_{ap} \Delta_p^{ij} \Delta_b^{ij} \\ &\quad - \sum_{ij} \delta_{ka} \Delta_b^{ij} \Gamma_i \kappa(x_i, x_j) \Gamma_j \Delta_\ell^{ij}. \end{aligned}$$

Multiplication of this term with some vector g_{ab} (resulting from stacking the elements of the $D \times d$ matrix g into a vector) requires storage of the $d \times N \times N$ array $R\Delta$ with elements $(R\Delta)_k^{ij}$, the $D \times N \times N$ array Δ with elements Δ_ℓ^{ij} , and the $N \times N$ matrix $\Gamma \Gamma^\top \otimes K$. Multiplication then takes the form

$$\begin{aligned} [H^1 g]_{k\ell} &= \sum_{j=1}^N \sum_{i=1}^N (R\Delta)_k^{ij} \Delta_\ell^{ij} \Gamma_i \Gamma_j \kappa_{x_i x_j} \\ &\quad \times \underbrace{\left[\sum_{a=1}^d (R\Delta)_a^{ij} \left[\sum_{b=1}^D \Delta_b^{ij} g_{ab} \right] \right]}_{\text{compute once in } \mathcal{O}(N^2 d D), \text{ store in } \mathcal{O}(N^2)}. \end{aligned} \quad (5)$$

Since the $N \times N$ matrix in the square brackets is independent of $k\ell$, it can be reused in the dD computations required to evaluate the full matrix–vector product, so the overall computation cost of this product is $\mathcal{O}(N^2 d D)$. The other required terms are of similar form. This means that approximate inversion of the Hessian, using an iterative solver like

the Lanczos or conjugate gradient methods, is achievable in time linear in D . The methods described here are computationally feasible even for high-dimensional problems. Our implementation of the active method, released along with this text, does not yet allow this kind of scalability, but the derivations above show that it is feasible in principle.

3 APPROXIMATE MARGINALIZATION OF GAUSSIAN PROCESS HYPERPARAMETERS

To turn the approximate Gaussian belief on R into an approximate Gaussian process belief on f , the active learning algorithm (constructed in Section 4) requires an (approximate) means of integrating over the belief on R . The elements of R form hyperparameters of the GP model. The problem of dealing with uncertainty in Gaussian process hyperparameters is a general one, also faced by other, non-active, Gaussian process regression models. This section presents a novel means of approximately integrating over the hyperparameters of a GP. The most widely used approach to learning GP hyperparameters is type-II maximum likelihood estimation (evidence maximization), or maximum *a posteriori* (MAP) estimation, which both approximate the likelihood as a delta function. However, ignoring the uncertainty in the hyperparameters in this way can lead to pathologies (MacKay, 2003).

For compact notation, all hyperparameters to be marginalized will be subsumed into a vector θ . We will denote as $m_{f|\mathcal{D},\theta}(x)$ the GP posterior mean prediction for $f(x)$ conditioned on data \mathcal{D} and θ , and similarly as $V_{f|\mathcal{D},\theta}(x)$ the posterior variance V of $f(x)$ conditioned on \mathcal{D} and θ .

We seek an approximation to the intractable posterior for $f_* = f(x_*)$, which requires marginalization over θ :

$$p(f_* | \mathcal{D}) = \int p(f_* | \mathcal{D}, \theta) p(\theta | \mathcal{D}) d\theta. \quad (6)$$

Assume a Gaussian conditional, $p(\theta|\mathcal{D}) = \mathcal{N}(\theta; \hat{\theta}, \Sigma)$, on the hyperparameters, such as the approximate distribution over R constructed in the preceding section. To make the integral in (6) tractable, we seek a linear approximation

$$p(f_*|\mathcal{D}, \theta) = \mathcal{N}(f_*; m_{f|\mathcal{D},\theta}(x_*), V_{f|\mathcal{D},\theta}(x_*)) \quad (7)$$

$$\simeq q(f_*; \theta) := \mathcal{N}(f_*; a^\top \theta + b, \nu^2), \quad (8)$$

using free parameters a, b, ν^2 to optimize the fit. The motivation for this approximation is that it yields a tractable marginal, $p(f_*|\mathcal{D}) \simeq \mathcal{N}(f_*; a^\top \hat{\theta} + b, \nu^2 + a^\top \Sigma a)$. Further, the posterior for θ typically has quite narrow width, over which $p(f_*|\mathcal{D}, \theta)$'s dependence on θ can be reasonably approximated. We choose the variables a, b, ν^2 by matching a local expansion of $q(f_* | \theta)$ to $p(f_*|\mathcal{D}, \theta)$. The expansion will be performed at $\theta = \hat{\theta}$, and at a $f_* = \hat{f}_*$ to be determined.

Specifically, we match as

$$\left. \frac{\partial}{\partial f_*} q(f_*; \theta) \right|_{\hat{\theta}, \hat{f}_*} = \left. \frac{\partial}{\partial f_*} p(f_*|\mathcal{D}, \theta) \right|_{\hat{\theta}, \hat{f}_*}; \quad (9)$$

$$\left. \frac{\partial}{\partial \theta_i} q(f_*; \theta) \right|_{\hat{\theta}, \hat{f}_*} = \left. \frac{\partial}{\partial \theta_i} p(f_*|\mathcal{D}, \theta) \right|_{\hat{\theta}, \hat{f}_*}; \quad (10)$$

$$\left. \frac{\partial^2}{\partial f_*^2} q(f_*; \theta) \right|_{\hat{\theta}, \hat{f}_*} = \left. \frac{\partial^2}{\partial f_*^2} p(f_*|\mathcal{D}, \theta) \right|_{\hat{\theta}, \hat{f}_*}; \quad (11)$$

$$\left. \frac{\partial^2}{\partial f_* \partial \theta_i} q(f_*; \theta) \right|_{\hat{\theta}, \hat{f}_*} = \left. \frac{\partial^2}{\partial f_* \partial \theta_i} p(f_*|\mathcal{D}, \theta) \right|_{\hat{\theta}, \hat{f}_*}. \quad (12)$$

An alternative set of constraints could be constructed by including second derivatives with respect to θ . But this would require computation scaling as $\mathcal{O}((\# \theta)^2)$, prohibitive for large numbers of hyperparameters, such as the $D \times d$ required to parameterize R for large D . We define

$$\hat{m} := m_{f|\mathcal{D}, \hat{\theta}} \quad \text{and} \quad \frac{\partial \hat{m}}{\partial \theta_i} := \left. \frac{\partial m_{f|\mathcal{D}, \theta}}{\partial \theta_i} \right|_{\theta = \hat{\theta}}, \quad (13)$$

along with analogous expressions for \hat{V} and $\frac{\partial \hat{V}}{\partial \theta_i}$. Turning to solving for a, b, ν^2 and f_* , note that, firstly, (9) implies that $a^\top \hat{\theta} + b = \hat{m}$, and that (11) implies that $\nu^2 = \hat{V}$. Rearranging (10) and (12), respectively, we have

$$2a_i = \frac{\partial \hat{V}}{\partial \theta_i} \left(\frac{1}{\hat{f}_* - \hat{m}} - \frac{\hat{f}_* - \hat{m}}{\hat{V}} \right) + 2 \frac{\partial \hat{m}}{\partial \theta_i}; \quad (14)$$

$$2a_i = 2 \frac{\partial \hat{V}}{\partial \theta_i} \frac{\hat{f}_* - \hat{m}}{\hat{V}} + 2 \frac{\partial \hat{m}}{\partial \theta_i}. \quad (15)$$

(14) and (15) can be solved only for

$$a_i = a_{i\pm} := \pm \frac{1}{\sqrt{3\hat{V}}} \frac{\partial \hat{V}}{\partial \theta_i} + \frac{\partial \hat{m}}{\partial \theta_i}; \quad (16)$$

$$f_* = \hat{f}_{*\pm} := \hat{m}(x_*) \pm \sqrt{\frac{\hat{V}(x_*)}{3}}. \quad (17)$$

In particular, note that the intuitive choice $f_* = \hat{m}(x_*)$, for which $\frac{\partial}{\partial f_*} p(f_*|\mathcal{D}, \theta) = 0$, gives q inconsistent constraints related to its variation with θ . Introducing the separation of $(\hat{V}(x_*)/3)^{1/2}$ provides optimal information about the curvature of $p(f_*|\mathcal{D}, \theta)$ with θ . Hence there are two possible values, $\hat{f}_{*\pm}$, to expand around, giving a separate Gaussian approximation for each. We average over the two solutions, giving an approximation that is a mixture of two Gaussians. We then further approximate this as a single moment-matched Gaussian.

The consequence of this approximation is that

$$p(f_* | \mathcal{D}) \simeq \mathcal{N}(f_*; \tilde{m}_{f|\mathcal{D}}(x_*), \tilde{V}_{f|\mathcal{D}}(x_*)), \quad (18)$$

where the marginal mean for f_* is $\tilde{m}_{f|\mathcal{D}}(x_*) := \hat{m}(x_*)$,

and the marginal variance is

$$\begin{aligned} \tilde{V}_{f|\mathcal{D}}(x_*) &:= \frac{4}{3} \hat{V}(x_*) + \frac{\partial \hat{m}(x_*)}{\partial \theta}^\top \Sigma \frac{\partial \hat{m}(x_*)}{\partial \theta} \\ &+ \frac{1}{3 \hat{V}(x_*)} \frac{\partial \hat{V}(x_*)}{\partial \theta}^\top \Sigma \frac{\partial \hat{V}(x_*)}{\partial \theta}. \end{aligned} \quad (19)$$

Figure 2 provides an illustration of our approximate marginal GP (henceforth abbreviated as MGP).

Our approach is similar to that of Osborne et al. (2012) (BBQ), for which $\tilde{V}_{f|\mathcal{D}} = V_{f|\mathcal{D}, \hat{\theta}} + \frac{\partial \hat{m}}{\partial \theta}^\top \Sigma \frac{\partial \hat{m}}{\partial \theta}$. However, BBQ ignores the variation of the predictive variance with changes in hyperparameters.

To compare the two methods, we generated (from a GP) $10 \times D$ random function values, \mathcal{D} , where D is the problem dimension. We then trained a GP with zero prior mean and ARD covariance on that data, and performed prediction for $10 \times D$ test data. Test points, (x_*, y_*) , were generated a small number (drawn from $\mathcal{U}(1, 3)$) of input scales away from a training point in a uniformly random direction. The MGP and BBQ were used to approximately marginalize over all GP hyperparameters (the output scale and D input scales), computing posteriors for the test points. We considered $D \in \{5, 10, 20\}$ and calculated the mean symmetrized Kullback–Leibler divergence (SKLD) over fifty random repetitions of each experiment. We additionally tested on two real datasets:⁴ yacht hydrodynamics (Gerritsma et al., 1981) and (centered) concrete compressive strength (Yeh, 1998). In these two, a random selection of 50 and 100 points, respectively, was used for training and the remainder for testing. All else was as above, with the exception that ten random partitions of each dataset were considered.

We evaluate performance using the SKLD between approximate posteriors and the “true” posterior (obtained using a run of slice sampling (Neal, 2003) with 10^5 samples and 10^4 burn-in); the better the approximate marginalization, the smaller this divergence. We additionally measured the average negative predictive log-likelihood, $-\mathbb{E}[\log p(y_* | x_*, \mathcal{D})]$, on the test points (x_*, y_*) . Results are displayed in Table 1; it can be seen that the MGP provides both superior predictive likelihoods and posteriors closer to the “true” distributions. The only exception is found on the yacht dataset, where the MGP’s SKLD score was penalized for having predictive variances that were consistently slightly larger than the “true” variances. However, these conservative variances, in better accommodating test points that were unexpectedly large or small, led to better likelihoods than the consistently over-confident MAP and BBQ predictions.

4 ACTIVE LEARNING OF GAUSSIAN PROCESS HYPERPARAMETERS

Now we turn to the question of actively selecting observation locations to hasten our learning of R . We employ an active learning strategy due to Houlsby et al. (2011), known as *Bayesian active learning by disagreement* (BALD). The idea is that, in selecting the location x of a function evaluation f to learn parameters θ , a sensible utility function is the expected reduction in the entropy of θ ,

$$v(x) := H(\Theta) - H(\Theta | F) = H(F) - H(F | \Theta), \quad (20)$$

also equal to the mutual information $I(\Theta; F)$ between f and θ . Mutual information, unlike differential entropies, is well-defined: the BALD objective is insensitive to changes in the representation of f and θ . The right-hand side of (20), the expected reduction in the entropy of f given the provision of θ , is particularly interesting. For our purposes, θ will parameterize $R \in \mathbb{R}^{d \times D}$; that is, θ is very high-dimensional, making the computation of $H(\Theta)$ computationally demanding. In contrast, the calculation of the entropy of $f \in \mathbb{R}$ is usually easy or even trivial. The right-hand side of (20) is particularly straightforward to evaluate under the approximation of Section 3, for which $p(f | \mathcal{D}, \theta)$ and the marginal $p(f | \mathcal{D})$ are both Gaussian. Further, under this approximation, $p(f | \mathcal{D}, \theta)$ has variance $\nu^2 = \hat{V}$ that is independent of θ , hence, $H(F | \Theta) = H(F | \Theta = \hat{\theta})$. We henceforth consider the equivalent but transformed utility function

$$v'(x) = \tilde{V}_{f|\mathcal{D}}(x) \left(V_{f|\mathcal{D}, \hat{\theta}}(x) \right)^{-1}. \quad (21)$$

The MGP approximation has only a slight influence on this objective – Figure 2 compares it to a full MCMC-derived marginal. With reference to (19), (21) encourages evaluations where the posterior mean and covariance functions are most sensitive to changes in θ (Figure 3), normalized by the variance in f : such points are most informative about the hyperparameters. An alternative to BALD is found in *uncertainty sampling*. Uncertainty sampling selects the location with highest variance, that is, its objective is simply $H(F)$, the first term in the BALD objective. This considers only the variance of a single point, whereas the BALD objective rewards points that assist in the learning of embeddings, thereby reducing the variance associated with all points. An empirical comparison of our method against uncertainty sampling follows below.

4.1 ACTIVE LEARNING OF LINEAR EMBEDDINGS FOR GAUSSIAN PROCESSES

To apply BALD to learning the linear embedding of a Gaussian process, we consider the case $R \subset \theta$; the GP hyperparameters define the embedding described in Section 2. Figure 4 demonstrates an example of active learning for the embedding of a two-dimensional function.

⁴<http://archive.ics.uci.edu/ml/datasets>.

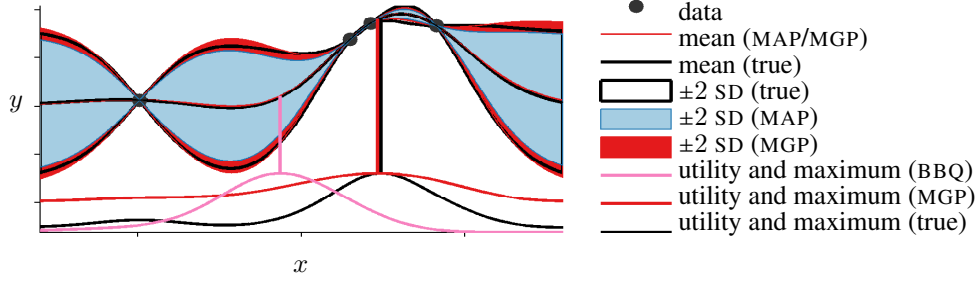


Figure 2: Approximate marginalization (MGP) of covariance hyperparameters θ increases the predictive variance to closely match the “true” posterior (obtained using slice sampling with 10^5 samples). BBQ (Osborne et al., 2012) provides a standard deviation differing from the MAP standard deviation by less than 3.1% everywhere, and would hence be largely invisible on this plot. The bottom of the figure displays the (normalized) mutual information $I(\Theta; F(x))$ (equal to the BALD utility function $v(x)$) for the various methods, and their maxima, giving the optimal positions for the next function evaluations. The MGP position is very close to the true position.

Table 1: Mean negative log-likelihood for test points and mean SKLD (nats) between approximate and true posteriors. Both metrics were averaged over test points, as well as over fifty and ten random repeats for synthetic and real experiments, respectively.

problem	dim	$-\mathbb{E}[\log p(y_* x_*, \mathcal{D})]$			SKLD		
		MAP	BBQ	MGP	MAP	BBQ	MGP
synthetic	5	3.58	2.67	1.73	0.216	0.144	0.0835
synthetic	10	3.57	3.10	1.86	0.872	0.758	0.465
synthetic	20	1.46	1.41	0.782	1.01	0.947	0.500
yacht	6	123.0	97.8	56.8	0.0322	0.0133	0.0323
concrete	8	$2.96 \cdot 10^9$	$2.96 \cdot 10^9$	$1.67 \cdot 10^9$	0.413	0.347	0.337

The latent model of lower dimension renders optimizing an objective with domain \mathcal{X} (e.g., $f(x)$, or the BALD objective) feasible even for high-dimensional \mathcal{X} . Instead of direct search over \mathcal{X} , one can choose a $u \in \mathcal{U}$, requiring search over only the low-dimensional \mathcal{U} , and then evaluate the objective at an $x \in \mathcal{X}$ for which $u = x R^\top$. A natural choice is the x which is most likely to actually map to u under R , that is, the x for which $p(u | x)$ is as tight as possible. For example, we could minimize $\log \det \text{cov}[u | x]$, subject to $\mathbb{E}[u | x] = x \hat{R}^\top$, by solving the appropriate program. For $d = 1$, this is a quadratic program that minimizes the variance $x \Sigma x^\top$ under the equality constraint. Finally, we evaluate the objective at the solution.

For simplicity, we will henceforth assume $\mathcal{X} = [-1, 1]^D$. For any box-bounded problem, there is an invertible affine transformation mapping the box to this \mathcal{X} ; this then requires only that R is composed with this transformation. Further, define the signature of the i th row of R to be $[\text{sign}(R_{i1}), \text{sign}(R_{i2}), \dots]$. Then, for the i th coordinate, the maximum and minimum value obtained by mapping the corners of \mathcal{X} through R are achieved by the corner matching this signature and its negative. This procedure defines the extreme corners of the search volume \mathcal{U} .

Consider the typical case in which we take $\tilde{\mu}$ as constant and $\tilde{\kappa}$ as isotropic (e.g., the exponentiated quadratic (1)). Since $p(f | X, R)$ is then invariant to orthogonal transformations of R in \mathbb{R}^d , there is no unique embedding. In the special case $d = 1$, R and $-R$ are equivalent. For most means and covariances there will be similar symmetries, and likely ever more of them as d increases.⁵ We therefore evaluate the performance of our algorithms not by comparing estimated to true R s, which is difficult due to these symmetries, but rather in the direct predictive performance for f .

4.2 ACTIVE LEARNING OF LINEAR EMBEDDINGS EXPERIMENTS

We now present the results of applying our proposed method for learning linear embeddings on both real and synthetic data with dimension up to $D = 318$. Given a function $f: \mathcal{X} \rightarrow \mathbb{R}$ with a known or suspected low-dimensional embedding, we compare the following methods for sequentially

⁵An alternative would be to place a prior on the *Stiefel manifold* rather than directly on R ; the Stiefel manifold accounts for most of these symmetries Minka (2000). This would require the rows of R to be orthogonal.

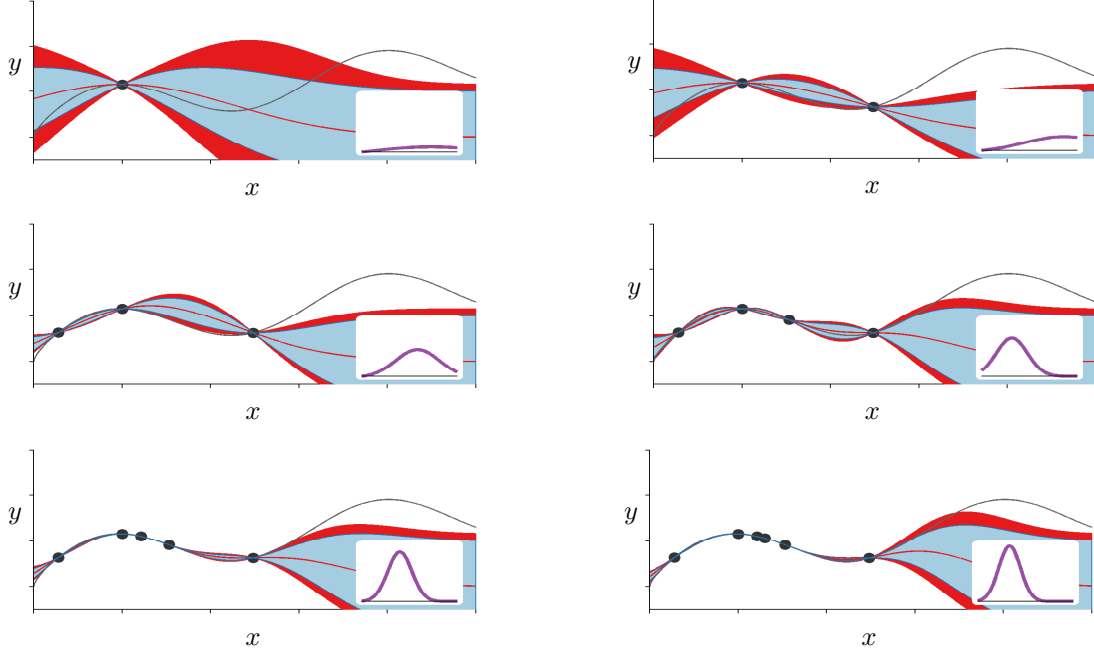


Figure 3: Active learning of the length scale of a one-dimensional GP (beginning at the top left and continuing across and then down): the next sample is taken where the MAP and approximate variances maximally disagree, normalized by the MAP variance. Samples are taken at a variety of separations to refine belief about the length scale. The inset plots (all of which share axes) display the approximate posteriors over log-length scales, which tighten with increasing numbers of samples. The legend is identical to that of Figure 2.

selecting $N = 100$ observations from the domain $[-1, 1]^D$: random sampling (RAND), a Latin hypercube design (LH), uncertainty sampling (UNC), and BALD. UNC and BALD use identical models (Laplace approximation on R followed by MGP) and hyperparameter priors. We also compare with LASSO, choosing the regularization parameter by minimizing squared loss on the training data. The functions that these methods are compared on are:

- Synthetic in-model data drawn from a GP matching our model with an embedding drawn from our prior, for $d \in \{2, 3\}$ and $D \in \{10, 20\}$.
- The Branin function, a popular test function for global optimization ($d = 2$), embedded in $D \in \{10, 20\}$ via an embedding drawn from our prior.
- The temperature data⁶ described in Snelson & Ghahramani (2006) ($D = 106$), with $d = 2$. The associated prediction problem concerns future temperature at a weather station, given the output of a circulation model. The training and validation points were combined to form the dataset, comprising 10 675 points.
- The normalized “communities and crime” (C&C)

⁶<http://theoval.cmp.uea.ac.uk/~gcc/competition>.

dataset from the UCI Machine Learning Repository⁷ ($D = 96$), with $d = 2$. The task here is to predict the number of violent crimes per capita in a set of US communities given historical data from the US Census and FBI. The LEMAS survey features were discarded due to missing values, as was a single record missing the “AsianPerCap” attribute, leaving 1 993 points.

- The “relative location of CT slices on axial axis” dataset from the UCI Machine Learning Repository⁸ ($D = 318$), with $d = 2$. The task is to use features extracted from slices of a CT scan to predict its vertical location in the human body. Missing features were replaced with zeros. Only axial locations in the range $[50, 60]$ were used. Features that did not vary over these points were discarded, leaving 3 071 points.

The CT slices and communities and crime datasets are, respectively, the highest- and third-highest-dimensional regression datasets available in the UCI Machine Learning Repository with real attributes; in second place is an unnormalized version of the C&C dataset.

⁷<http://archive.ics.uci.edu/ml/datasets/Communities+and+Crime>.

⁸<http://archive.ics.uci.edu/ml/datasets/Relative+location+of+CT+sllices+on+axial+axis>.

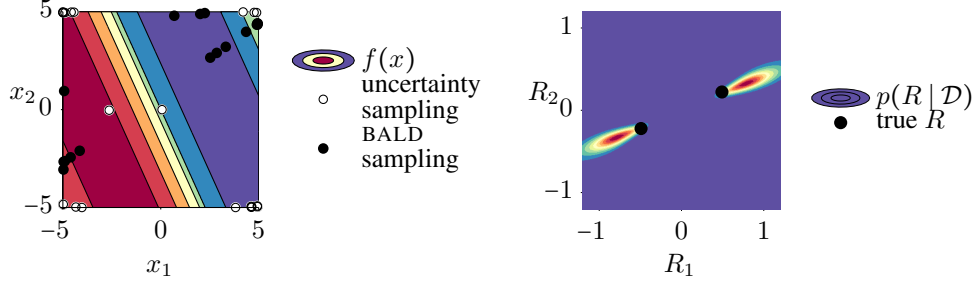


Figure 4: Left: Twenty samples selected by uncertainty sampling and BALD for a function f with a one-dimensional linear embedding. Note that uncertainty sampling prefers corner locations, to maximize the variance in u , $x\Sigma x^\top$, and hence the variance in f ; these points are less useful for learning f and its embedding than those selected by BALD. Right: the posterior over embeddings returned by the BALD samples, concentrated near the true embedding (equivalent under negation).

Table 2: The combination of MGP and BALD actively learns embeddings whose predictive performance improves on alternatives. Average negative log predictive probability and average RMSE on test functions for various D and d .

dataset	D/d	$-\mathbb{E}[\log p(y_* x_*, \hat{R})]$				RMSE				
		RAND	LH	UNC	BALD	RAND	LH	UNC	BALD	LASSO
synthetic	10/2	0.272	0.224	-0.564	-0.649	0.412	0.371	0.146	0.138	0.842
synthetic	10/3	0.711	0.999	0.662	0.465	0.553	0.687	0.557	0.523	0.864
synthetic	20/2	0.804	0.745	0.749	0.470	0.578	0.549	0.551	0.464	0.853
synthetic	20/3	1.07	1.10	1.04	0.888	0.714	0.740	0.700	0.617	0.883
Branin	10/2	3.87	3.90	1.58	0.0165	18.2	17.8	3.63	2.29	40.0
Branin	20/2	4.00	3.70	3.55	3.63	18.3	14.8	13.4	15.0	39.1
communities & crime	96/2	1.09	—	1.17	1.01	0.720	—	0.782	0.661	1.16
temperature	106/2	0.566	—	0.583	0.318	0.423	—	0.427	0.328	0.430
CT slices	318/2	1.30	—	1.26	1.16	0.878	—	0.845	0.767	0.900

For the synthetic and Branin problems, where the true embedding R was chosen explicitly, we report averages over five separate experiments differing only in the choice of R . On these datasets, the UNC and BALD methods selected points by successively maximizing their respective objectives on a set of 20 000 fixed points in the input domain, 10 000 selected uniformly in $[-1, 1]^D$ and 10 000 selected uniformly in the unit D -sphere. For a given D , these points were fixed across methods and experimental runs. This choice allows us to compare methods based only on their objectives and not the means of optimizing them.

For the real datasets (temperature, communities and crimes, and CT slices), each method selected from the available points; LH is incapable of doing so and so is not considered on these datasets. The real datasets were further processed by transforming all features to the box $[-1, 1]^D$ via the “subtract min, divide by max” map and normalizing the outputs to have zero mean and unit variance. For the synthetic problems, we added i.i.d. Gaussian observation noise with variance $\sigma^2 = (0.1)^2$. For the remaining problems, the datapoints were used directly (assuming that these real measurements already reflect noise).

After each method selected 100 observations, we compare the quality of the learned embeddings by fixing the hyper-

parameters of a GP to the MAP embedding at termination and measuring predictive performance. This is intended to emulate a fixed-budget embedding learning phase followed by an experiment using only the most likely R . We chose $N = 100$ training points and 1 000 test points uniformly at random from those available; these points are common to all methods. We report root-mean-square error (RMSE) and the average negative predictive log-likelihood on the test points. The RMSE measures predictive accuracy, whereas the log-likelihood additionally captures the accuracy of variance estimates. This procedure was repeated 10 times for each experiment; the reported numbers are averages.

The embedding prior $p(R)$ was set to be i.i.d. zero-mean Gaussian with standard deviation $5/4 D^{-1}$. This choice roughly implies that we expect $[-1, 1]^D$ to map approximately within $[-2.5, 2.5]^d$, a box five length scales on each side, under the unknown embedding. This prior is extremely diffuse and does not encode any structure of R beyond preferring low-magnitude values. At each step, the mode of the log posterior over R was found using using L-BFGS, starting from both the previous best point and one random restart drawn from $p(R)$.

The results are displayed in Table 2. The active algorithm achieves the most accurate predictions on all but one prob-

lem, including each of the real datasets, according to both metrics. These results strongly suggest an advantage for actively learning linear embeddings.

5 CONCLUSIONS

Active learning in regression tasks should include hyperparameters, in addition to the function model itself. Here we studied simultaneous active learning of the function and a low-dimensional linear embedding of its input domain. We also developed a novel means of approximately integrating over the hyperparameters of a GP model. The resulting algorithm addresses needs in a number of domains, including Bayesian optimization, Bayesian quadrature, and also the underlying idea of nonparametric Gaussian regression itself. Empirical evaluation demonstrates the efficacy of the resulting algorithm on both synthetic and real problems in up to 318 dimensions, and an analysis of computational cost shows that the algorithm can, at least in principle, be scaled to problems of much larger dimensionality as well.

ACKNOWLEDGMENTS

Part of this work was supported by the German Science Foundation (DFG) under reference number ‘GA 1615/1–1.’

References

- Bergstra, J. and Bengio, Y. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305, 2012.
- Brochu, E., Cora, V. M., and de Freitas, N. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*, 2010.
- Candes, E. and Tao, T. The Dantzig selector: Statistical estimation when p is much larger than n . *Annals of Statistics*, 35(6):2313–2351, 2007.
- Carpentier, A. and Munos, R. Bandit theory meets compressed sensing for high dimensional stochastic linear bandit. *arXiv preprint arXiv:1205.4094*, 2012.
- Chen, B., Castro, R., and Krause, A. Joint optimization and variable selection of high-dimensional Gaussian processes. In *Proceedings of the 29th Annual International Conference on Machine Learning*, 2012.
- Gerritsma, J., Onnink, R., and Versluis, A. Geometry, resistance and stability of the delft systematic yacht hull series. *International Shipbuilding Progress*, 28(328):276–297, 1981.
- Girard, A. and Murray-Smith, R. Gaussian processes: Prediction at a noisy input and application to iterative multiple-step ahead forecasting of time-series. In *Switching and Learning in Feedback Systems*, number 3355 in Lecture Notes in Computer Science. Springer, 2005.
- Guestrin, C., Krause, A., and Singh, A. P. Near-optimal sensor placements in Gaussian processes. In *Proceedings of the 22nd Annual International Conference on Machine Learning*, pp. 265–272. ACM, 2005.
- Hennig, P. and Kiefel, M. Quasi-Newton methods – A new direction. In *Proceedings of the 29th Annual International Conference on Machine Learning*, 2012.
- Houlsby, N., Huszár, F., Ghahramani, Z., and Lengyel, M. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011.
- Hutter, F. *Automated configuration of algorithms for solving hard computational problems*. PhD thesis, University of British Columbia, 2009.
- Hutter, F., Hoos, H., and Leyton-Brown, K. Sequential model-based optimization for general algorithm configuration. In *Learning and Intelligent Optimization*, pp. 507–523. Springer, 2011.
- Iwata, T., Houlsby, N., and Ghahramani, Z. Active learning for interactive visualization. *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics (AISTATS 2013)*, (31), 2013.
- Lawrence, N. D. A unifying probabilistic perspective for spectral dimensionality reduction: Insights and new models. *Journal of Machine Learning Research*, 13:1609–1638, 2012.
- MacKay, D. J. C. Bayesian interpolation. *Neural Computation*, 4(3):415–447, 1992a.
- MacKay, D. J. C. Information-based objective functions for active data selection. *Neural Computation*, 4(4):590–604, 1992b.
- MacKay, David J. C. *Information theory, inference and learning algorithms*. Cambridge University Press, 2003.
- McHutchon, A. and Rasmussen, C. E. Gaussian process training with input noise. In *Advances in Neural Information Processing Systems 24*, 2011.
- Minka, T. Automatic choice of dimensionality for PCA. Technical Report 514, MIT Media Laboratory, 2000.
- Neal, R. M. *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto, 1995.
- Neal, R. M. Slice sampling. *Annals of Statistics*, 31(3):705–767, 2003.
- O’Hagan, A. Bayes-Hermite quadrature. *Journal of Statistical Planning and Inference*, 29:245–260, 1991.
- Osborne, M. A., Duvenaud, D., Garnett, R., Rasmussen, C. E., Roberts, S. J., and Ghahramani, Z. Active learning of model evidence using Bayesian quadrature. In *Advances in Neural Information Processing Systems 25*, pp. 46–54, 2012.
- Rasmussen, C. E. and Williams, C. K. I. *Gaussian Processes for Machine Learning*. MIT Press, 2006.
- Snelson, E. and Ghahramani, Z. Variable noise and dimensionality reduction for sparse Gaussian processes. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence (UAI 2006)*, pp. 461–468, 2006.

- Tibshirani, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.
- Wang, Z., Zoghi, M., Hutter, F., Matheson, D., and de Freitas, N. Bayesian optimization in a billion dimensions via random embeddings. *arXiv preprint arXiv:1301.1942*, 2013.
- Williams, C. K. I. and Rasmussen, C. E. Gaussian processes for regression. In *Advances in Neural Information Processing Systems* 8, 1996.
- Yeh, I-C. Modeling of strength of high-performance concrete using artificial neural networks. *Cement and Concrete Research*, 28 (12):1797–1808, 1998.

Estimating Causal Effects by Bounding Confounding

Philipp Geiger, Dominik Janzing, Bernhard Schölkopf

Max Planck Institute for Intelligent Systems, Tübingen, Germany
{pgeiger, janzing, bs}@tuebingen.mpg.de

Abstract

Assessing the causal effect of a treatment variable X on an outcome variable Y is usually difficult due to the existence of unobserved common causes. Without further assumptions, observed dependences do not even prove the existence of a causal effect from X to Y . It is intuitively clear that strong statistical dependences between X and Y do provide evidence for X influencing Y if the influence of common causes is known to be weak. We propose a framework that formalizes effect versus confounding in various ways and derive upper/lower bounds on the effect in terms of a priori given bounds on confounding. The formalization includes information theoretic quantities like information flow and causal strength, as well as other common notions like effect of treatment on the treated (ETT). We discuss several scenarios where upper bounds on the strength of confounding can be derived. This justifies to some extent human intuition which assumes the presence of causal effect when strong (e.g. close to deterministic) statistical relations are observed.

1 INTRODUCTION

In many situations one wants to estimate the causal effect from an observable X to an observable Y , e.g. if/to what extent smoking causes lung cancer. It is widely agreed that randomized experiments constitute the gold standard for inferring the causal effect. The reason for this is that an ideal randomized experiment excludes the possibility of a (partially) unobserved confounding cause U . However, in many cases conducting randomized experiments would be very expensive or impossible. In these cases, if we do not have

any additional knowledge on the setting, then inference of the (precise or approximate) causal effect is generally deemed impossible. In case we have additional knowledge however, it may be possible to *estimate* the causal effect, i.e. derive (upper and/or lower) *bounds* for it. For instance it is well known that if we observe an instrumental variable Z together with X and Y , those bounds can be derived (see e.g. [Pearl, 2000]).

1.1 THE FORMAL FRAMEWORK

To make our discussion as precise as possible we will from this point on use the framework for causality developed in [Pearl, 2000]. Particularly we will make use of the do-calculus formalizing interventions on variables. (It should be mentioned though that we slightly deviate from Pearl’s definition of the do-operator as we will further explicate in Section 2.1.)

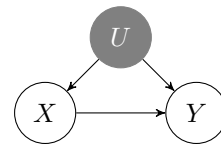


Figure 1: Causal DAG for the confounding scenario (gray means unobserved).

The causal DAG in Figure 1 formalizes the causal structure underlying U, X, Y . Note that we allow U and in some cases also X, Y to be multivariate. Furthermore keep in mind that in some scenarios discussed in the paper we assume U to be partially observed. Our general goal is to estimate the causal effect from X to Y . Formally, this means that we want to estimate $P(Y|\text{do } X=x)$ or related quantities such as the effect of treatment on the treated (ETT) [Pearl, 2000] or the causal strength from X to Y [Janzing et al., 2013]. Without further assumptions, these quantities are impossible to estimate. To give an extreme example, one can imagine observing the deterministic

relationship $P(y|x) = \delta_{yx}$, with δ_{yx} denoting the Kronecker delta. This observation can be induced by two completely different underlying causal structures, the first one being that Y in fact is produced by copying X , the second one being that both X and Y are copied from U without X having any causal effect on Y .

1.2 RELATED WORK

Several approaches have been developed to identify or estimate causal effects in spite of hidden confounders.

Back-door/front-door criterion (see [Pearl, 2000, 2009]): This approach applies for the case where some variables on the confounding path or between X and Y are measured and we know the causal structure underlying all variables together. There are several criteria that allow to decide whether the causal effect from X to Y is identifiable. Furthermore, formulas are available to calculate the effect in these cases. Besides the natural limitation namely requiring a lot of information on additional variables and structures, one drawback of this method is that it cannot be used if X is deterministically coupled to the back-door variable.

Instrumental variables (see e.g. [Pearl, 2000, Angrist et al., 1996, Efron and Feldman, 1991]): In the simplest case, the causal DAG in Figure 1 is augmented by a parentless node Z with an arrow to X . An important example are clinical trials with partial compliance. The additional Z allows to infer bounds on the average causal effect. One drawback of this method is that it yields a convex optimization problem with the number of equations growing exponentially with the cardinality of X . Furthermore, to apply this method one needs to know $p(X, Y|Z)$ while in Section 4.2 we present a scenario where $p(Z)$ (additional to $p(X, Y)$) helps to estimate the causal effect.

Regression discontinuity design (see e.g. [Thistlewaite and Campbell, 1960, Imbens and Lemieux, 2008, Lee and Lemieux, 2010]): This framework is applicable to cases where an additional observable Z mediating between U and X is measured and X is a deterministic function of Z that contains a discontinuity. Under the assumption of linearity of the remaining structural equations, the effect from X to Y , i.e. the linear coefficient, can be identified. One limitation of this method is that it needs the discontinuity and a high slope alone does not suffice.

1.3 OUR APPROACH

The approach we suggest to estimate causal effects in spite of confounding consists of two parts: In the *first part* (Section 3) we propose various possibilities to formalize the following notions:

- *Observed dependence*: the dependence of Y on X that we can observe based on $p(X, Y)$.
- *Back-door dependence*: the “spurious association” [Pearl, 2000] between X and Y due to the confounder U .
- *Causal effect*: what happens to Y upon intervening on X ; this includes notions of conditional causal effect such as the ETT.

For all formalizations we present inequalities (see Table 1 for an overview) which turn out to always have the following prototypical form:

$$\left[\begin{array}{c} \text{back-door} \\ \text{dependence} \end{array} \right] \geq d \left(\left[\begin{array}{c} \text{observed} \\ \text{dependence} \end{array} \right], \left[\begin{array}{c} \text{causal} \\ \text{effect} \end{array} \right] \right)$$

(where the $d(\cdot, \cdot)$ stands for deviation measure). In some of these results, observed dependence, back-door dependences, and causal effect are real numbers and $d(\cdot, \cdot)$ simply stands for the usual difference which allows us to convert the prototypical form into

$$\left[\begin{array}{c} \text{causal} \\ \text{effect} \end{array} \right] \geq \left[\begin{array}{c} \text{observed} \\ \text{dependence} \end{array} \right] - \left[\begin{array}{c} \text{back-door} \\ \text{dependence} \end{array} \right],$$

which may be more convenient for applications.

In order to draw conclusions on the true causal effect using the inequalities from the first part, one needs to have knowledge on the back-door dependence. Therefore, in the *second part* (Section 4), we demonstrate how in various situations one can come up with bounds on the back-door dependence. Based on these together with the observed dependence one can then infer bounds on the true causal effect.

Before getting started, in Section 2, we present several definitions which are needed throughout the paper.

2 PREREQUISITES

Keep in mind the following definitions and results throughout the paper.

2.1 DEFINITION OF CAUSAL MODEL AND do-OPERATOR

As already mentioned we essentially use the framework developed in [Pearl, 2000] to discuss causal relations. Let $V = \{X_1, \dots, X_n\}$ be a set of random variables. A *causal model* M w.r.t. V consists of a DAG G , noise variables N_i for each i with a joint distribution P that makes them jointly independent, and structural equations $X_i := f_i(\text{PA}_i^G, N_i)$ for each i , where PA_i^G denotes the parents of X_i in G .

Table 1: Formalizing observed dependence (O), back-door dependence (B), causal effect (C) and deviation measure (d). $\mathfrak{C}_{X \rightarrow Y}$ denotes the strength of the influence of X on Y in the sense of [Janzing et al., 2013], $I(X \rightarrow Y | \text{do } U)$ is the information flow by [Ay and Polani, 2008], $D[\cdot || \cdot]$ denotes Kullback-Leibler divergence [Cover and Thomas, 1991]. The symbol do is the do-operator defined by [Pearl, 2000].

Sec.	O/B/C/d	formalized by ...
3.1	O	$I(X : Y)$
	B	$I(U : X), \mathfrak{C}_{U \rightarrow X}$
	C	$\mathfrak{C}_{X \rightarrow Y}$
	d	difference
3.2	O	$I(X : Y)$
	B	$I(U : X)$
	C	$I(X \rightarrow Y \text{do } U)$
	d	difference
3.3	O	$p(Y X=x)$
	B	$I(X : U), \min\{\mathfrak{C}_{U \rightarrow X}, \mathfrak{C}_{U \rightarrow Y}\}$
	C	$p(Y \text{do } X=x)$
	d	$D[\cdot \cdot]$
3.4	O	$\mathbb{E}[\text{d}_x \log p(Y X=x)^2]$
	B	$\mathbb{E}[\partial_2 \log p(Y X=x, \text{do } X=x)^2]$
	C	$\mathbb{E}[\partial_1 \log p(Y X=x, \text{do } X=x)^2]$
	d	difference
3.5	O	$\mathbb{E}[Y X=x'] - \mathbb{E}[Y X=x]$
	B	$\mathbb{E}[Y X=x', \text{do } X=x]$ $-\mathbb{E}[Y X=x, \text{do } X=x]$
	C	$\mathbb{E}[Y X=x', \text{do } X=x']$ $-\mathbb{E}[Y X=x', \text{do } X=x]$
	d	difference
3.6	O	$\text{d}_x \mathbb{E}[Y X=x]$
	B	$\partial_1 \mathbb{E}[Y X=x, \text{do } X=x]$
	C	$\partial_2 \mathbb{E}[Y X=x, \text{do } X=x]$
	d	difference

Now we define the *do-operator*. Given any $X_i \in V$ the post-intervention causal model $M_{\text{do } X_i=x'}$ is obtained from M in the following way: For each child X_j of X_i , we replace the structural equation $X_j = f_j(\text{PA}_j^G \setminus X_i, X_i, N_j)$ by $X_j = f_j(\text{PA}_j^G \setminus X_i, x', N_j)$. Note that the random variable X_i stays in the model it just no longer has children. (This is the point where we deviate from [Pearl, 2000]. Note the analogy to the splitting of nodes in [Richardson and Robins, 2013, Robins et al., 2007], where X_i is replaced with two deterministically coupled variables, one being adjacent to the parents of X_i and one to the children of X_i . Then we refer to an intervention on the latter one while the first one is kept.)

The new set of structural equations of $M_{\text{do } X_i=x'}$

together with the noise variables N_i for all i induce a new joint distribution on X_1, \dots, X_n which we denote by $P(X_1, \dots, X_n | \text{do } X=x')$. In particular, given M contains the variables X and Y , quantities such as $P(Y|X=x, \text{do } X=x')$ are well defined. (Note that $P(Y|X=x, \text{do } X=x')$, based on our definition of $M_{\text{do } X=x'}$, coincides with the counterfactual distribution $P(Y_{x'}|X=x)$ as defined in [Pearl, 2000].)

2.2 DISTRIBUTIONS AND DENSITIES

Throughout the paper we will work with U, X, Y with discrete as well as with continuous ranges.

Unless noted otherwise we make the following fundamental assumption regarding the distributions of the random variables in a causal model M with causal DAG G : for each $X_j \in V$, the random variable $f_j(\text{pa}_j, N_j)$ has a density w.r.t. the Lebesgue measure (in the continuous case) or w.r.t. the counting measure (in the discrete case) respectively, denoted by $q_j(x_j; \text{pa}_j)$ for each value pa_j of PA_j (note that we have to slightly deviate from this assumption in Section 4.1 though). This assumption implies the following simple lemma, which is only formulated for the case $n = 3$, since we only need this case in the present paper. A proof can be found in the supplement.

Lemma 1. *Under the assumption made above, the joint distribution of X_1, X_2, X_3 induced by a causal model M or any post-interventional model $M_{\text{do } X_i=x}$ has a density w.r.t. the Lebesgue measure (in the continuous case) or counting measure (in the discrete case), respectively. Moreover, this density factorizes according to the causal DAG belonging to the respective model.*

Regarding any causal model M with causal DAG as depicted in Figure 1, also the following simple statement holds true. The proof is obvious but we present it in the supplement anyway. Note that the lemma is similar to [Pearl, 2000, Corollary 7.3.2], but better tailored for our definition of $M_{\text{do } X=x}$.

Lemma 2. *For all x we have*

$$p(Y|X=x, \text{do } X=x) = p(Y|X=x),$$

$$\mathbb{E}[Y|X=x, \text{do } X=x] = \mathbb{E}[Y|X=x].$$

3 THE RELATION BETWEEN OBSERVED DEPENDENCE, BACK-DOOR DEPENDENCE AND CAUSAL EFFECT

In this section we present various possibilities to formalize the notions of observed dependence, back-door dependence and causal effect. For all formalizations

we prove that the back-door dependence is equal to or upper bounds the deviation between the observed dependence and the actual causal effect.

Subsections 3.1, 3.2 apply to X, Y, U with finite range. Subsections 3.3, 3.5 apply to X, Y, U with arbitrary range. Subsections 3.4 and 3.6 apply to X with continuous range.

Keep in mind that $H(\cdot)$ denotes the Shannon entropy, $I(\cdot : \cdot)$ ($I(\cdot : \cdot | \cdot)$) the (conditional) mutual information, and $D[\cdot \| \cdot]$ the Kullback-Leibler divergence, all based on logarithms with base 2. For details see [Cover and Thomas, 1991].

3.1 ESTIMATING THE CAUSAL STRENGTH FROM X TO Y

The basic quantities in this section are:

- observed dep.: $I(X : Y)$,
- back-door dep.: $I(X : U)$, $\mathfrak{C}_{U \rightarrow X}$,
- causal effect: $\mathfrak{C}_{X \rightarrow Y}$.

We consider the case of U, X, Y having finite range. [Janzing et al., 2013] proposed a definition for the causal strength of a set of arrows in a causal DAG.

We briefly want to repeat this definition for the special case of measuring the strength of a single arrow. For a set of observables $V = \{X_1, \dots, X_n\}$, a DAG G' with V as the set of nodes and a joint distribution $p(X_1, \dots, X_n)$ and for any arrow $X_i \rightarrow X_j$ in G' we first define the distribution $p_{X_i \rightarrow X_j}$ corresponding to deleting $X_i \rightarrow X_j$ from the graph and feeding X_j with an independent copy of X_i instead, see also [Ay and Krakauer, 2007]:

$$p_{X_i \rightarrow X_j}(x_j | \text{pa}_{X_j}^{X_i \rightarrow X_j}) := \sum_{x'_i} p(x'_i) p(y | x'_i, \text{pa}_{X_j}^{X_i \rightarrow X_j}),$$

$$p_{X_i \rightarrow X_j}(x_k | \text{pa}_{X_k}^{X_i \rightarrow X_j}) := p(x_k | \text{pa}_{X_k}), \text{ for all } k \neq j,$$

$$p_{X_i \rightarrow X_j}(x_1, \dots, x_n) := \prod_{k=1}^n p_{X_i \rightarrow X_j}(x_k | \text{pa}_{X_k}^{X_i \rightarrow X_j}),$$

where $\text{pa}_{X_k}^{X_i \rightarrow X_j}$ denotes (values of) the set of parents of X_k in the modified graph G' without arrow $X_i \rightarrow X_j$ (obviously this actually only makes a change for pa_{X_j}). Now we are able to define the *causal strength* $\mathfrak{C}_{X_i \rightarrow X_j}$ by the impact of the edge deletion:

$$\mathfrak{C}_{X_i \rightarrow X_j} := D[p(X_1, \dots, X_n) \| p_{X_i \rightarrow X_j}(X_1, \dots, X_n)].$$

Let us get back to our specific confounding scenario (the causal DAG in Figure 1). For general DAGs, [Janzing et al., 2013] shows $\mathfrak{C}_{X \rightarrow Y} \geq I(X : Y | PA_Y \setminus X)$, that is, the information Y contains about X given its other parents is a lower bound for causal

strength (they argue that this property would be desirable for other information-theoretic measures of causal strength as well). Hence in our confounding scenario (Figure 1) we have $\mathfrak{C}_{X \rightarrow Y} \geq I(X : Y | U)$. Also keep in mind that $\mathfrak{C}_{U \rightarrow X} = I(U : X)$ in our setting.

Lemma 3. *We have*

$$I(X : Y | U) \geq I(X : Y) - I(X : U). \quad (1)$$

Proof. The statement follows from the fact that $I(X : Y | U) + I(X : U) = I(X : U, Y) \geq I(X : Y)$. \square

We consider $I(X : Y)$ as a measure of *observed dependence* between X and Y . The following theorem shows that the *back-door dependence* $\mathfrak{C}_{U \rightarrow X}$ bounds the difference between the observed dependence and the true *causal effect* $\mathfrak{C}_{X \rightarrow Y}$.

Theorem 1. *We have*

$$\mathfrak{C}_{U \rightarrow X} \geq I(X : Y) - \mathfrak{C}_{X \rightarrow Y}. \quad (2)$$

Proof. This follows from Lemma 3 together with the fact that $\mathfrak{C}_{X \rightarrow Y} \geq I(X : Y | U)$ and $\mathfrak{C}_{U \rightarrow X} = I(X : U)$ in our confounding scenario (i.e. the DAG in Figure 1). \square

3.2 ESTIMATING THE INFORMATION FLOW FROM X TO Y

The basic quantities in this section are:

- observed dep.: $I(X : Y)$,
- back-door dep.: $I(X : U)$,
- causal effect: $I(X \rightarrow Y | \text{do } U)$.

Another information theoretic quantity to measure the causal effect of X on Y is the *information flow* proposed by [Ay and Polani, 2008]. In our setting (the causal DAG in Figure 1) it is defined as

$$\begin{aligned} I(X \rightarrow Y | \text{do } U) &:= \\ &\sum_u p(u) \sum_x p(x | \text{do } U=u) \sum_y p(y | \text{do } X=x, \text{do } U=u) \\ &\times \log \frac{p(y | \text{do } X=x, \text{do } U=u)}{\sum_{x'} p(y | \text{do } X=x', \text{do } U=u) p(x' | \text{do } U=u)}. \end{aligned}$$

Since $p(y | \text{do } X=x, \text{do } U=u) = p(y | x, u)$ in our setting, we simply have $I(X \rightarrow Y | \text{do } U) = I(X : Y | U)$.

So we can establish a theorem for the information flow similar to the one for the causal strength. it follows immediately from Lemma 3.

Theorem 2. *We have*

$$I(X : U) \geq I(X : Y) - I(X \rightarrow Y | \text{do } U). \quad (3)$$

3.3 BOUNDING THE KULLBACK-LEIBLER DIVERGENCE BETWEEN $p(Y|X=x)$ AND $p(Y|\text{do } X=x)$

The basic quantities in this section are:

- observed dep.: $p(Y|X=x)$,
- back-door dep.: $I(X : U)$, $\min\{\mathfrak{C}_{U \rightarrow X}, \mathfrak{C}_{U \rightarrow Y}\}$,
- causal effect: $p(Y|\text{do } X=x)$.

In some sense, $p(Y|\text{do } X=x)$ is the most fundamental characterization of the *causal effect* from X to Y , while $p(Y|X=x)$ can be seen as the corresponding characterization of their *observed dependence*. In this section we show that the deviation between these two objects can be bounded by quantities which measure the *back-door dependence*, $I(X : U)$ and $\min\{\mathfrak{C}_{U \rightarrow X}, \mathfrak{C}_{U \rightarrow Y}\}$. We formalize the notion of deviation here by

$$\begin{aligned} D[p(Y|X) \| p(Y|\text{do } X)] \\ := \sum_x p(x) D[p(Y|x) \| p(Y|\text{do } X=x)]. \end{aligned}$$

Theorem 3. *We have*

$$\begin{aligned} D[p(Y|X) \| p(Y|\text{do } X)] &\leq \min\{\mathfrak{C}_{U \rightarrow X}, \mathfrak{C}_{U \rightarrow Y}\} \\ &\leq I(X : U). \end{aligned}$$

Proof. First note that

$$\begin{aligned} p_{U \rightarrow X}(u, x, y) &= p(u)p(x)p(y|u, x) \text{ and} \\ p_{U \rightarrow Y}(u, x, y) &= p(u)p(x|u) \sum_{u'} p(y|u, x)p(u'). \end{aligned}$$

This implies

$$\begin{aligned} p(y|\text{do } X=x) &= p_{U \rightarrow X}(y|X=x) \text{ and} \\ p(y|\text{do } X=x) &= p_{U \rightarrow Y}(y|X=x). \end{aligned}$$

Therefore, using the chain rule for Kullback-Leibler divergence,

$$\begin{aligned} D[p(Y|X) \| p(Y|\text{do } X)] &= D[p(Y|X) \| p_{U \rightarrow X}(Y|X)] \\ &\leq D[p(X, Y) \| p_{U \rightarrow X}(X, Y)] = \mathfrak{C}_{U \rightarrow X} (= I(U : X)). \end{aligned}$$

Similarly one can derive $D[p(Y|X) \| p(Y|\text{do } X)] \leq \mathfrak{C}_{U \rightarrow Y}$. \square

The above theorem makes a statement w.r.t. the divergence between $p(Y|x)$ and $p(Y|\text{do } X=x)$ averaged over all values x of X . But it is also possible to derive a pointwise version:

Theorem 4. *For all x*

$$D[p(Y|x) \| p(Y|\text{do } x)] \leq D[p(U|x) \| p(U)],$$

with equality iff $p(u|x) = p(u)$ for all u .

Proof. By the log sum inequality we have

$$\begin{aligned} p(y|x) \log \frac{p(y|x)}{p(y|\text{do } x)} \\ &= \left(\sum_u p(y|x, u)p(u|x) \right) \log \frac{\sum_u p(y|x, u)p(u|x)}{\sum_u p(y|x, u)p(u)} \\ &\leq \sum_u p(y|x, u)p(u|x) \log \frac{p(y|x, u)p(u|x)}{p(y|x, u)p(u)} \\ &= \sum_u p(y, u|x) \log \frac{p(u|x)}{p(u)}. \end{aligned} \quad (4)$$

Equality holds in (4) iff $p(y|x, u)p(u|x) = cp(y|x, u)p(u)$ for all u and some constant c , i.e. iff $p(u|x) = p(u)$ for all u . Summing over all y yields the claimed inequality. \square

Note that taking the average w.r.t. X in Theorem 4 is another way to prove the first part of Theorem 3. With a similar proof we can also derive the following inequality w.r.t. the “inverse mutual information” $D[p(U)p(X) \| p(U, X)]$ (as opposed to the usual mutual information $I(U : X) = D[p(U, X) \| p(U)p(X)]$). For this purpose let us define

$$\begin{aligned} D[p(Y|\text{do } X) \| p(Y|X)] \\ := \sum_x p(x) \sum_y p(y|\text{do } X=x) \log \frac{p(y|\text{do } X=x)}{p(y|X=x)}. \end{aligned}$$

Corollary 1. *We have*

$$D[p(Y|\text{do } X) \| p(Y|X)] \leq D[p(U)p(X) \| p(U, X)].$$

To assess which bound is relevant for a scenario, we recall that for two distributions p and q , $D[p \| q]$ diverges when $q = 0$ and $p > 0$ on a set of Lebesgue measure greater than 0. If the observed dependence $p(Y|X)$ is deterministic, $p(Y|\text{do } X)$ needs to be deterministic if $D[p(Y|\text{do } X) \| p(Y|X)]$ is finite.

3.3.1 An example for bounding the average causal effect from X to Y

Often one is interested in estimating the *average causal effect* $\mathbb{E}[Y|\text{do } X=x'] - \mathbb{E}[Y|\text{do } X=x]$ for two values x, x' of X [Pearl, 2000], in particular because this quantity is easy to interpret. In what follows, we want to give an example how one can derive bounds on this quantity based on Theorem 3. It is important to mention however, that the assumptions we make are very restrictive. The purpose of the example is only to show that information theoretic bounds on the back-door dependence *can*, under appropriate assumptions, imply bounds for the average causal effect.

Let X be binary, $p(Y|x) = \mathcal{N}(\mu_x, \sigma^2)$, and $p(Y|\text{do } X=x) = \mathcal{N}(\mu_{\text{do } x}, \sigma_{\text{do}}^2)$, for $x = 0, 1$ (hence particularly $\mathbb{E}[Y|\text{do } X=x] = \mu_{\text{do } x}$).¹

In this case we can calculate (have in mind that \ln is the natural logarithm)

$$\begin{aligned} & p(X=0)(\mu_0 - \mu_{\text{do } 0})^2 + p(X=1)(\mu_1 - \mu_{\text{do } 1})^2 \\ &= 2\sigma_{\text{do}}^2 \left(D[p(Y|X) \| p(Y|\text{do } X)] - \ln \frac{\sigma_{\text{do}}^2}{\sigma^2} - \frac{\sigma^2}{2\sigma_{\text{do}}^2} + \frac{1}{2} \right) \\ &\leq 2\sigma_{\text{do}}^2 \left(\min\{\mathfrak{C}_{U \rightarrow X}, \mathfrak{C}_{U \rightarrow Y}\} - \ln \frac{\sigma_{\text{do}}^2}{\sigma^2} - \frac{\sigma^2}{2\sigma_{\text{do}}^2} + \frac{1}{2} \right). \end{aligned} \quad (5)$$

Now assume we fix $\min\{\mathfrak{C}_{U \rightarrow X}, \mathfrak{C}_{U \rightarrow Y}\}$ and σ_{do}^2 . Keep in mind that μ_0, μ_1, σ^2 are observed. Then we can derive upper and lower bounds on the average causal effect $\mu_{\text{do } 1} - \mu_{\text{do } 0}$ by maximizing and minimizing it, respectively, under the constraints on the pair $(\mu_{\text{do } 1}, \mu_{\text{do } 0})$ imposed by inequality (5).

3.4 ESTIMATING THE FISHER INFORMATION

The basic quantities in this section are:

- observed dep.: $\mathcal{F}_{Y|X}(x)$,
- back-door dep.: $\mathcal{F}_{Y|X, \text{do } X}^1(x, x)$,
- causal effect: $\mathcal{F}_{Y|X, \text{do } X}^2(x, x)$.

In the following, $\partial_i f(x, x')$, $i = 1, 2$, denotes the partial derivative w.r.t. the i th argument of f evaluated at position (x, x') . And $\text{d}_x g(x)$ denotes the total derivative of $g(x)$ w.r.t. x at position x , in particular $\text{d}_x f(x, x) = \text{d}_x g(x)$ for $g(x) := f(x, x)$.

Given a family of distributions depending on continuous parameters, *Fisher information* provides a natural way to quantify the sensitivity of a probability distribution to infinitesimal parameter changes. It plays an important role for the error made when estimating the true parameter value from empirical data [Rao, 1945]. Here we quantify causal influence by the sensitivity of $p(Y|\text{do } x)$ to small changes of x . This can be considered as a “local” measure of causal strength in the neighborhood of x . We introduce the following notation:

$$\begin{aligned} \mathcal{F}_{Y|X}(x) &:= \int (\text{d}_x \log p(y|X=x))^2 p(y|X=x) dy, \\ \mathcal{F}_{Y|X, \text{do } X}^i(x, x') &:= \\ &\int (\partial_i \log p(y|X=x, \text{do } X=x'))^2 p(y|X=x, \text{do } X=x') dy, \end{aligned}$$

¹Note, however, that both $p(Y|X=0)$ and $p(Y|X=1)$ being Gaussian actually provides some evidence for the absence of confounding since a confounder will often destroy this simple structure of $P(Y|X)$ [Janzing et al., 2011].

for $i = 1, 2$.

Theorem 5. *For all x*

$$\sqrt{\mathcal{F}_{Y|X}(x)} - \sqrt{\mathcal{F}_{Y|X, \text{do } X}^2(x, x)} \leq \sqrt{\mathcal{F}_{Y|X, \text{do } X}^1(x, x)}.$$

A proof can be found in the supplement.

3.5 ESTIMATING THE EFFECT OF TREATMENT ON THE TREATED FROM X TO Y

The basic quantities in this section are:

- observed dep.: $\mathbb{E}[Y|X=x'] - \mathbb{E}[Y|X=x]$,
- back-door dep.: $\mathbb{E}[Y|X=x', \text{do } X=x] - \mathbb{E}[Y|X=x, \text{do } X=x]$,
- causal effect: $\mathbb{E}[Y|X=x', \text{do } X=x'] - \mathbb{E}[Y|X=x', \text{do } X=x]$.

Following [Pearl, 2000], we define the quantity

$$\mathbb{E}[Y|X=x', \text{do } X=x'] - \mathbb{E}[Y|X=x', \text{do } X=x]$$

as the *effect of treatment on the treated*. As the name already suggests, the idea behind this quantity is to measure the deviation between the average response of the treated subjects and the average response of these same subjects had they not been treated. The following result w.r.t. the effect of treatment on the treated follows from Lemma 2.

Theorem 6. *We have for all x, x'*

$$\begin{aligned} & \mathbb{E}[Y|X=x'] - \mathbb{E}[Y|X=x] \\ &= \mathbb{E}[Y|X=x', \text{do } X=x'] - \mathbb{E}[Y|X=x', \text{do } X=x] \\ &\quad + \mathbb{E}[Y|X=x', \text{do } X=x] - \mathbb{E}[Y|X=x, \text{do } X=x]. \end{aligned}$$

Note that in mediation analysis [Pearl, 2001, Avin et al., 2005, Robins and Greenland, 1992] a similar splitting into direct and indirect effect is used. However mediation analysis addresses the problem of defining direct and indirect *causal* effects and not back-door dependences.

We briefly want to discuss the other quantities that appear in the theorem. Obviously, $\mathbb{E}[Y|X=x'] - \mathbb{E}[Y|X=x]$ measures the *observed dependence* of Y on X . Now keep in mind that in $M_{\text{do } X=x}$, X has no causal effect on Y anymore and hence Y statistically depends on X solely via U . Therefore the difference

$$\mathbb{E}[Y|X=x', \text{do } X=x] - \mathbb{E}[Y|X=x, \text{do } X=x]$$

measures the strength of the *back-door dependence* of Y on X .

3.6 ESTIMATING THE DIFFERENTIAL EFFECT OF TREATMENT ON THE TREATED FROM X TO Y

The basic quantities in this section are:

- observed dep.: $d_x \mathbb{E}[Y|X=x]$,
- back-door dep.: $\partial_1 \mathbb{E}[Y|X=x, \text{do } X=x]$,
- causal effect: $\partial_2 \mathbb{E}[Y|X=x, \text{do } X=x]$.

First note that by $\partial_i \mathbb{E}[Y|X=x, \text{do } X=x']$ we mean $\partial_i f(x, x')$ for $f(x, x') := \mathbb{E}[Y|X=x, \text{do } X=x']$ (recall that ∂_i denotes the partial derivative w.r.t. the i th argument). In the case of continuous random variables U, X, Y we want to consider the following quantity (if it exists i.e. if the conditional expectation is differentiable)

$$\partial_2 \mathbb{E}[Y|X=x, \text{do } X=x],$$

which we call *differential effect of treatment on the treated* or simply *differential effect* in cases where this does not lead to confusions. It is the analog to the discrete effect of treatment on the treated (see Section 3.5) for the case of infinitesimal interventional changes on X ; we simply replaced a difference by a derivative.

Similar to Theorem 6 we can establish the following result. It follows from the chain rule for derivatives together with Lemma 2.

Theorem 7. *For all x*

$$\begin{aligned} d_x \mathbb{E}[Y|X=x] &= \partial_1 \mathbb{E}[Y|X=x, \text{do } X=x] \\ &\quad + \partial_2 \mathbb{E}[Y|X=x, \text{do } X=x]. \end{aligned}$$

The interpretation of this theorem is similar to the one for Theorem 6. Obviously, $d_x \mathbb{E}[Y|X=x]$ is the *observed dependence*, whereas the quantity $\partial_1 \mathbb{E}[Y|X=x, \text{do } X=x]$ measures the *back-door dependence* of Y on X . So the observed dependence of Y on X splits into the causal effect plus the back-door dependence.

4 PROTOTYPICAL SCENARIOS WITH BOUNDS ON THE BACK-DOOR DEPENDENCE

In this section we present several prototypical scenarios where bounds on the back-door dependence between X and Y can be derived. Together with our results from Section 3 these bounds help to estimate causal effect from X to Y .

4.1 A QUALITATIVE TOY EXAMPLE

We want to give an example that demonstrates how human intuition concerning observed dependence and causal effect relates to the theorems from Section 3.

Assume there is a drug that is indicated for a specific disease. We observe some not too small number of people with the disease and see that some of them take the drug and some do not. We find that all persons who took the drug recovered on the same day whereas none of the persons not taking the drug recovered that fast. For each sick person let X denote the date he or she takes the drug and Y the date he or she recovers. Since these are only observations, we cannot exclude that there is a confounder U , i.e. we assume the usual causal DAG (Figure 1). We estimate the distribution of Y given X by the empirical distribution, i.e. $p(y|x) = \delta_{yx}$, where δ_{yx} denotes the Kronecker delta.

Given the above setting probably most people would argue that there has to be some effect from the drug to the immediate healing of those people who took it. However, formally and without further assumptions $p(Y|x)$ alone does not even tell us if there is a causal link from X to Y at all. With the help of Theorem 3 though, we can formally reason as follows. We make the weak additional assumption that X cannot be completely determined by U which we formalize by $I(U : X) < H(X)$. It seems implausible that there exists a common cause of X and Y that determines both, the exact date X a person takes the drug and the recovering date Y . E.g. the wealth of a person may strongly influence both, the treatment he or she takes and how quickly he or she recovers (via the general health condition), however it seems not plausible that the wealth determines the exact day of taking the drug and of recovering.

For a proof by contradiction we may assume that there is no causal effect from X to Y , i.e. $p(Y|\text{do } X=x) = p(Y|\text{do } X=x')$ for all x, x' . Then

$$\begin{aligned} D[p(Y|X)||p(Y|\text{do } X=x)] &= \sum_x p(x) D[p(Y|X=x)||p(Y|\text{do } X=x)] \\ &= \sum_x p(x) \sum_y \delta_{yx} \log \frac{\delta_{yx}}{P(Y=y|\text{do } X=x)} \\ &= \sum_x p(x) \log \frac{1}{p(Y=x|\text{do } X=0)} \geq H(X), \end{aligned}$$

where the last inequality is due to Gibb's inequality [Cover and Thomas, 1991].

On the other hand, due to Theorem 3 we have

$$D[p(Y|X)||p(Y|\text{do } X)] \leq I(X : U) < H(X),$$

which yields the contradiction. Hence we could formally show that there has to be some causal effect from X to Y , $p(Y|\text{do } X=x) \neq p(Y|\text{do } X=x')$ for some x, x' . Note that the above argumentation completely

transfers to any other situation where $p(y|x) = \delta_{yx}$, particularly any other range of X and Y .

4.2 PARTIAL RANDOMIZATION SCENARIO

We first discuss a formal scenario, then an application example, and afterwards we discuss how the scenario and our result is related to the instrumental variable design [Pearl, 2000].

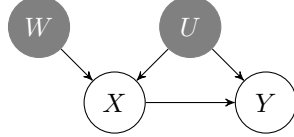


Figure 2: The partial randomization causal DAG.

4.2.1 THE FORMAL PROTOTYPE

We consider a scenario where we have measured X and Y , and where hidden variables U and W are present and we know the distribution of W . The underlying causal structure of all variables looks like the causal DAG depicted in Figure 2. We assume that W is binary. Furthermore we assume that in this scenario $I(U : X|W = 0) = 0$. The intuition behind this assumption is that W decides whether X is influenced by U ($W = 1$) or not. This scenario implies the following inequality. A proof can be found in the supplement.

Proposition 1. *In the given scenario we have $I(U : X) \leq H(X)p(W=1)$.*

Now we can employ our results from Sections 3.1 through 3.3. We obtain the following bounds:

$$I(X : Y) - \mathfrak{C}_{X \rightarrow Y} \leq H(X)p(W=1), \quad (6)$$

$$I(X : Y) - I(X \rightarrow Y|\text{do } U) \leq H(X)p(W=1), \quad (7)$$

$$D[p(Y|X) \| p(Y|\text{do } X)] \leq H(X)p(W=1). \quad (8)$$

Note that under strong assumptions, one can also apply the result from Section 3.3.1 to estimate the average causal effect $\mathbb{E}[Y|\text{do } X=x'] - \mathbb{E}[Y|\text{do } X=x]$ for two values x, x' of X .

4.2.2 ADVERTISEMENT LETTER EXAMPLE

Assume we are managers of a mail order company, and want to know the effect of sending advertisement letters on the ordering behavior of the recipients. We have a data set of (X, Y) pairs with X denoting whether a letter was sent to a specific person and let Y denote the total costs of the products ordered by this

person afterwards (within some fixed time span). Assume we have enough data to estimate $p(X, Y)$. Furthermore, assume that so far there were already imperfect guidelines based on rough intuition on whom to send letters and whom not. These guidelines introduce a potential confounder U since letters were more likely sent to potential customers with properties that made them also more likely to order something (if the guidelines were not complete nonsense). It is known however that only some employees stuck to these guidelines. Let W denote whether a letter was sent out in compliance with these guidelines ($W = 1$) or not.

Based on an estimate of how many employees complied with the guidelines, we also have an estimate of $p(W = 1)$, i.e. the fraction of letters that was sent out in compliance with the guidelines. Based on Proposition 1, we know that $I(U : X) \leq H(X)p(W=1)$. Hence we have an upper bound on the back-door dependence of Y on X . Particularly we can apply inequalities (6) to (8) and, under strong additional assumptions, the result w.r.t. the average causal effect from Section 3.3.1.

For example by (6) we have $I(X : Y) - H(X)p(W=1) \leq \mathfrak{C}_{X \rightarrow Y}$. Now assume $H(X) \approx 1$ (we sent a letter to roughly every second person in our register) and $p(W=1) \approx 0.5$ (only half the employees stuck to the guidelines). Then if we observe a strong dependence of Y on X , say $I(X : Y) \approx 0.75$, then we can conclude that $\mathfrak{C}_{X \rightarrow Y} \gtrsim 0.25$, i.e. our advertisement letters have a significant effect on the potential customers.

4.2.3 DIFFERENCE TO INSTRUMENTAL VARIABLE DESIGN

We already mentioned the instrumental variable design [Pearl, 2000] in Section 1. In this design it is assumed that an additional variable W is observed such that the causal structure of all variables together is as depicted in Figure 2, except that W is not hidden. The prototypical application scenario for this design are clinical trials with partial compliance. [Pearl, 2000] describes a method to derive bounds on the average causal effect $\mathbb{E}[Y|\text{do } X=1] - \mathbb{E}[Y|\text{do } X=0]$. This analysis heavily depends on the range of X , Y , and W and involves convex optimization in 15-dimensional space already for the case where all variables are binary (since U can be assumed to attain 16 different values).

The advantage of our approach lies in the fact that the ranges of the variables may be arbitrary without increasing the complexity – for the cost of getting less tight bounds than an explicit modeling, of course. One can get bounds for the case where neither X nor

Y are binary, e.g., in a drug testing scenario with different doses and descriptions of health conditions that are more complex than just reporting recovery or not. Moreover, we do not need complete knowledge of $p(Y, X|W)$ provided that we have some knowledge on W that provides upper bounds on $I(X:U)$.

4.3 A VARIANT OF THE REGRESSION DISCONTINUITY DESIGN

We already mentioned the regression discontinuity design (RDD) [Thistlewaite and Campbell, 1960, Imbens and Lemieux, 2008, Lee and Lemieux, 2010] in Section 1. It is a quasi-experimental design that can help to estimate the causal effect from X to Y in cases where an additional variable Z is measured and the underlying causal DAG of all variables together is as depicted in Figure 3. The design usually requires that X is a deterministic function of Z that contains a discontinuity, that all remaining structural equations are linear, and that $\mathbb{E}[U|Z = z]$ is continuous in z . (Note that the causal DAG in Figure 3 is a special case of the general confounding scenario depicted in Figure 1, which can be seen by replacing U in Figure 1 by $U' := (U, Z)$.)

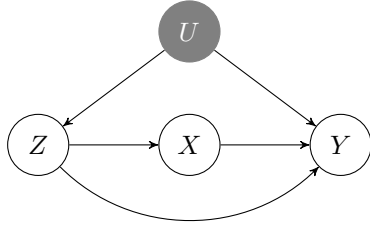


Figure 3: The causal DAG for the RDD and our variant of it.

We now want to consider a scenario inspired by the RDD, which allows to bound the back-door dependence in the sense of Section 3.6 and thus makes Theorem 7 applicable to estimate the causal effect $\partial_2 \mathbb{E}[Y|X=x, \text{do } X=x]$, i.e. the differential effect of treatment on the treated. The scenario differs from the RDD in that neither a discontinuity in the structural equation for X , nor linearity of the remaining structural equations is required.

Assume the causal DAG in Figure 3. Furthermore assume that $X = f_X(Z)$ for a function f_X that is differentiable. (This is the point where our scenario differs from RDD.) Suppose f_X is invertible, $g := f_X^{-1}$. It can easily be seen that this implies

$$\begin{aligned} \partial_1 \mathbb{E}[Y|X=x, \text{do } X=x] \\ = \partial_1 \mathbb{E}[Y|Z=g(x), \text{do } X=x]g'(x). \end{aligned}$$

Note that $\partial_1 \mathbb{E}[Y|Z=g(x), \text{do } X=x]$ means the deriva-

tive of $\mathbb{E}[Y|Z=z, \text{do } X=x]$ w.r.t. z at position $(g(x), x)$. Applying Theorem 7 yields

$$\begin{aligned} d_x \mathbb{E}[Y|X=x] - \partial_2 \mathbb{E}[Y|X=x, \text{do } X=x] \\ = \partial_1 \mathbb{E}[Y|Z=g(x), \text{do } X=x]g'(x). \end{aligned}$$

Hence if for any position x_0 of X we assume a bound on the strength of the “back-door” dependence of Y on Z , $\partial_1 \mathbb{E}[Y|Z=g(x_0), \text{do } X=x_0]$, and if $|g'(x_0)|$ is comparably small (which is the case when $|f'_X(g(x_0))|$ is big), then we can bound the difference between observed dependence and causal effect at position x_0 .

For instance, if we consider the observed dependence $d_x \mathbb{E}[Y|X=x]$ as a realistic scale based on which one can constrain $\partial_1 \mathbb{E}[Y|Z=g(x), \text{do } X=x]$, formally

$$|\partial_1 \mathbb{E}[Y|Z=g(x), \text{do } X=x]| \leq c |d_x \mathbb{E}[Y|X=x]|,$$

for some c , then one can bound the modulus of the causal effect from below:

$$\begin{aligned} |\partial_2 \mathbb{E}[Y|Z=g(x), \text{do } X=x]| \\ \geq (1 - c|g'(x)|) |d_x \mathbb{E}[Y|X=x]|. \end{aligned}$$

Obviously one weakness of the above argument is that the estimation of the causal effect heavily depends on the bound that we assume w.r.t. the “back-door” dependence of Y on Z , $\partial_1 \mathbb{E}[Y|Z=g(x), \text{do } X=x]$. However, this can be seen as a quantitative analogon to the qualitative assumption of the RDD that $\mathbb{E}[U|Z = z]$ is continuous in z .

Keep in mind that our results on Fisher information (Section 3.4) can be used in the case where X is not a deterministic function of Z that changes rapidly but instead the conditional probability $p(X|z)$ changes fast at some $z = z_0$.

5 CONCLUSIONS

In this paper, we analyzed a simple intuition linking observation and causation: if the observed dependence is strong and the effect of confounding is known to be weak, then we can infer a causal effect. We did this by employing a number of different notions for measuring dependence and causation, leading to different theoretical bounds. We do not argue that at present, there is a single formalization that best captures all aspects of this intuition, rather, we try to shed light on properties of the various notions by applying them to the same fundamental problem. While bounding confounding appears easier based on information theoretic quantities, expressing the influence from the treatment to the outcome variable by e.g. the effect of treatment on the treated (ETT) seems more relevant for practical purposes. We discussed several prototypical scenarios where bounds on confounding can be derived.

References

- J. Angrist, G. Imbens, and D. Rubin. Identification of Causal Effects Using Instrumental Variables. *Journal of the American Statistical Association*, 91(434): 444–455, 1996.
- C. Avin, I. Shpitser, and J. Pearl. Identifiability of path-specific effects. In *Proceedings of the International Joint Conference in Artificial Intelligence*, pages 357–363, Edinburgh, Scotland, 2005.
- N. Ay and D. Krakauer. Geometric robustness and biological networks. *Theory in Biosciences*, 125:93–121, 2007.
- N. Ay and D. Polani. Information flows in causal networks. *Advances in Complex Systems*, 11(1):17–41, 2008.
- T. Cover and J. Thomas. *Elements of Information Theory*. Wileys Series in Telecommunications, New York, 1991.
- B. Efron and D. Feldman. Compliance as an Explanatory Variable in Clinical Trials. *Journal of the American Statistical Association*, 86(413):9–17, 1991.
- G. Imbens and T. Lemieux. Regression discontinuity designs: A guide to practice. *Journal of Econometrics*, 142:615–635, 2008.
- D. Janzing, E. Sgouritsa, O. Stegle, P. Peters, and B. Schölkopf. Detecting low-complexity unobserved causes. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence (UAI 2011)*, 2011.
- D. Janzing, D. Balduzzi, M. Grosse-Wentrup, and B. Schölkopf. Quantifying causal influences. *Annals of Statistics*, 41(5):2324–2358, 2013.
- D. Lee and T. Lemieux. Regression Discontinuity Designs in Economics. *Journal of Economic Literature*, 48:281–355, 2010.
- J. Pearl. *Causality*. Cambridge University Press, 2000.
- J. Pearl. Direct and indirect effects. In *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 411–420, San Francisco, CA, 2001. Morgan Kaufmann.
- J. Pearl. Causal inference in statistics: An overview. *Statistics Surveys*, 3:96–146, 2009.
- R. C. Rao. Information and the accuracy attainable in the estimation of statistical parameters. *Bull. Calcutta Math. Soc.*, 37:81–91, 1945. ISSN 0008-0659.
- T. Richardson and J. Robins. Single world intervention graphs (swigs). Technical report, University of Washington, 2013.
- J. Robins and S. Greenland. Identifiability and exchangeability for direct and indirect effects. *Epidemiology*, 3(2):143–155, 1992.
- J. Robins, T. VanderWeele, and T. Richardson. Discussion of ”causal effects in the presence of non compliance a latent variable interpretation” by forcina, a. *Metron*, LXIV(3):288–298, 2007.
- D. Thistlewaite and D. Campbell. Regression-discontinuity analysis: an alternative to the ex-post facto experiment. *Journal of Educational Psychology*, 51:309–317, 1960.

Bayesian Optimization with Unknown Constraints

Michael A. Gelbart
Harvard University
Cambridge, MA

Jasper Snoek
Harvard University
Cambridge, MA

Ryan P. Adams
Harvard University
Cambridge, MA

Abstract

Recent work on Bayesian optimization has shown its effectiveness in global optimization of difficult black-box objective functions. Many real-world optimization problems of interest also have constraints which are unknown *a priori*. In this paper, we study Bayesian optimization for constrained problems in the general case that noise may be present in the constraint functions, and the objective and constraints may be evaluated independently. We provide motivating practical examples, and present a general framework to solve such problems. We demonstrate the effectiveness of our approach on optimizing the performance of online latent Dirichlet allocation subject to topic sparsity constraints, tuning a neural network given test-time memory constraints, and optimizing Hamiltonian Monte Carlo to achieve maximal effectiveness in a fixed time, subject to passing standard convergence diagnostics.

1 INTRODUCTION

Bayesian optimization (Mockus et al., 1978) is a method for performing global optimization of unknown “black box” objectives that is particularly appropriate when objective function evaluations are expensive (in any sense, such as time or money). For example, consider a food company trying to design a low-calorie variant of a popular cookie. In this case, the design space is the space of possible recipes and might include several key parameters such as quantities of various ingredients and baking times. Each evaluation of a recipe entails computing (or perhaps actually measuring) the number of calories in the proposed cookie. Bayesian optimization can be used to propose new candidate recipes such that good results are found with few evaluations.

Now suppose the company also wants to ensure the *taste* of the cookie is not compromised when calories are reduced.

Therefore, for each proposed low-calorie recipe, they perform a taste test with sample customers. Because different people, or the same people at different times, have differing opinions about the taste of cookies, the company decides to require that at least 95% of test subjects must like the new cookie. This is a constrained optimization problem:

$$\min_{\mathbf{x}} c(\mathbf{x}) \text{ s.t. } \rho(\mathbf{x}) \geq 1 - \epsilon,$$

where \mathbf{x} is a real-valued vector representing a recipe, $c(\mathbf{x})$ is the number of calories in recipe \mathbf{x} , $\rho(\mathbf{x})$ is the fraction of test subjects that like recipe \mathbf{x} , and $1 - \epsilon$ is the minimum acceptable fraction, i.e., 95%.

This paper presents a general formulation of constrained Bayesian optimization that is suitable for a large class of problems such as this one. Other examples might include tuning speech recognition performance on a smart phone such that the user’s speech is transcribed within some acceptable time limit, or minimizing the cost of materials for a new bridge, subject to the constraint that all safety margins are met.

Another use of constraints arises when the search space is known *a priori* but occupies a complicated volume that cannot be expressed as simple coordinate-wise bounds on the search variables. For example, in a chemical synthesis experiment, it may be known that certain combinations of reagents cause an explosion to occur. This constraint is not unknown in the sense of being a discovered property of the environment as in the examples above—we do not want to discover the constraint boundary by trial and error explosions of our laboratory. Rather, we would like to specify this constraint using a boolean noise-free oracle function that declares input vectors as valid or invalid. Our formulation of constrained Bayesian optimization naturally encapsulates such constraints.

1.1 BAYESIAN OPTIMIZATION

Bayesian optimization proceeds by iteratively developing a global statistical model of the unknown objective function. Starting with a prior over functions and a likelihood, at each

iteration a posterior distribution is computed by conditioning on the previous evaluations of the objective function, treating them as observations in a Bayesian nonlinear regression. An *acquisition function* is used to map beliefs about the objective function to a measure of how promising each location in input space is, if it were to be evaluated next. The goal is then to find the input that maximizes the acquisition function, and submit it for function evaluation.

Maximizing the acquisition function is ideally a relatively easy proxy optimization problem: evaluations of the acquisition function are often inexpensive, do not require the objective to be queried, and may have gradient information available. Under the assumption that evaluating the objective function is expensive, the time spent computing the best next evaluation via this inner optimization problem is well spent. Once a new result is obtained, the model is updated, the acquisition function is recomputed, and a new input is chosen for evaluation. This completes one iteration of the Bayesian optimization loop.

For an in-depth discussion of Bayesian optimization, see Brochu et al. (2010b) or Lizotte (2008). Recent work has extended Bayesian optimization to multiple tasks and objectives (Krause and Ong, 2011; Swersky et al., 2013; Zuluaga et al., 2013) and high dimensional problems (Wang et al., 2013; Djolonga et al., 2013). Strong theoretical results have also been developed (Srinivas et al., 2010; Bull, 2011; de Freitas et al., 2012). Bayesian optimization has been shown to be a powerful method for the meta-optimization of machine learning algorithms (Snoek et al., 2012; Bergstra et al., 2011) and algorithm configuration (Hutter et al., 2011).

1.2 EXPECTED IMPROVEMENT

An acquisition function for Bayesian optimization should address the exploitation vs. exploration tradeoff: the idea that we are interested both in regions where the model believes the objective function is low (“exploitation”) and regions where uncertainty is high (“exploration”). One such choice is the Expected Improvement (EI) criterion (Mockus et al., 1978), an acquisition function shown to have strong theoretical guarantees (Bull, 2011) and empirical effectiveness (e.g., Snoek et al., 2012). The expected improvement, $EI(\mathbf{x})$, is defined as the expected amount of improvement over some target t , if we were to evaluate the objective function at \mathbf{x} :

$$EI(\mathbf{x}) = \mathbb{E}[(t - y)_+] = \int_{-\infty}^{\infty} (t - y)_+ p(y | \mathbf{x}) dy, \quad (1)$$

where $p(y | \mathbf{x})$ is the predictive marginal density of the objective function at \mathbf{x} , and $(t - y)_+ \equiv \max(0, t - y)$ is the improvement (in the case of minimization) over the target t . EI encourages both exploitation and exploration because it is large for inputs with a low predictive mean (exploitation) and/or a high predictive variance (exploration). Of-

ten, t is set to be the minimum over previous observations (e.g., Snoek et al., 2012), or the minimum of the expected value of the objective (Brochu et al., 2010a). Following our formulation of the problem, we use the minimum expected value of the objective such that the probabilistic constraints are satisfied (see Section 1.5, Eq., 6).

When the predictive distribution under the model is Gaussian, EI has a closed-form expression (Jones, 2001):

$$EI(\mathbf{x}) = \sigma(\mathbf{x}) (z(\mathbf{x})\Phi(z(\mathbf{x})) + \phi(z(\mathbf{x}))) \quad (2)$$

where $z(\mathbf{x}) \equiv \frac{t - \mu(\mathbf{x})}{\sigma(\mathbf{x})}$, $\mu(\mathbf{x})$ is the predictive mean at \mathbf{x} , $\sigma^2(\mathbf{x})$ is the predictive variance at \mathbf{x} , $\Phi(\cdot)$ is the standard normal CDF, and $\phi(\cdot)$ is the standard normal PDF. This function is differentiable and fast to compute, and can therefore be maximized with a standard gradient-based optimizer. In Section 3 we present an acquisition function for constrained Bayesian optimization based on EI.

1.3 OUR CONTRIBUTIONS

The main contribution of this paper is a general formulation for constrained Bayesian optimization, along with an acquisition function that enables efficient optimization of such problems. Our formulation is suitable for addressing a large class of constrained problems, including those considered in previous work. The specific improvements are enumerated below.

First, our formulation allows the user to manage uncertainty when constraint observations are noisy. By reformulating the problem with probabilistic constraints, the user can directly address this uncertainty by specifying the required confidence that constraints are satisfied.

Second, we consider the class of problems for which the objective function and constraint function need not be evaluated jointly. In the cookie example, the number of calories might be predicted very cheaply with a simple calculation, while evaluating the taste is a large undertaking requiring human trials. Previous methods, which assume joint evaluations, might query a particular recipe only to discover that the objective (calorie) function for that recipe is highly unfavorable. The resources spent simultaneously evaluating the constraint (taste) function would then be very poorly spent. We present an acquisition function for such problems, which incorporates this user-specified cost information.

Third, our framework, which supports an arbitrary number of constraints, provides an expressive language for specifying arbitrarily complicated restrictions on the parameter search spaces. For example if the total memory usage of a neural network must be within some bound, this restriction could be encoded as a separate, noise-free constraint with very low cost. As described above, evaluating this low-cost constraint would take priority over the more expensive con-

straints and/or objective function.

1.4 PRIOR WORK

There has been some previous work on constrained Bayesian optimization. Gramacy and Lee (2010) propose an acquisition function called the integrated expected conditional improvement (IECI), defined as

$$\text{IECI}(\mathbf{x}) = \int_{\mathcal{X}} [\text{EI}(\mathbf{x}') - \text{EI}(\mathbf{x}'|\mathbf{x})] h(\mathbf{x}') d\mathbf{x}'. \quad (3)$$

In the above, $\text{EI}(\mathbf{x}')$ is the expected improvement at \mathbf{x}' , $\text{EI}(\mathbf{x}'|\mathbf{x})$ is the expected improvement at \mathbf{x}' given that the objective has been observed at \mathbf{x} (but without making any assumptions about the observed value), and $h(\mathbf{x}')$ is an arbitrary density over \mathbf{x}' . In words, the IECI at \mathbf{x} is the expected reduction in EI at \mathbf{x}' , under the density $h(\mathbf{x}')$, caused by observing the objective at \mathbf{x} . Gramacy and Lee use IECI for constrained Bayesian optimization by setting $h(\mathbf{x}')$ to the probability of satisfying the constraint. This formulation encourages evaluations that inform the model in places that are likely to satisfy the constraint.

Zuluaga et al. (2013) propose the Pareto Active Learning (PAL) method for finding Pareto-optimal solutions when multiple objective functions are present and the input space is a discrete set. Their algorithm classifies each design candidate as either Pareto-optimal or not, and proceeds iteratively until all inputs are classified. The user may specify a confidence parameter determining the tradeoff between the number of function evaluations and prediction accuracy. Constrained optimization can be considered a special case of multi-objective optimization in which the user’s utility function for the “constraint objectives” is an infinite step function: constant over the feasible region and negative infinity elsewhere. However, PAL solves different problems than those we intend to solve, because it is limited to discrete sets and aims to classify each point in the set versus finding a single optimal solution.

Snoek (2013) discusses constrained Bayesian optimization for cases in which constraint violations arise from a failure mode of the objective function, such as a simulation crashing or failing to terminate. The author introduces the weighted expected improvement acquisition function, namely expected improvement weighted by the predictive probability that the constraint is satisfied at that input.

1.5 FORMALIZING THE PROBLEM

In Bayesian optimization, the objective and constraint functions are in general unknown for two reasons. First, the functions have not been observed everywhere, and therefore we must interpolate or extrapolate their values to new inputs. Second, our observations may be noisy; even after multiple observations at the same input, the true function is

not known. Accounting for this uncertainty is the role of the model, see Section 2.

However, before solving the problem, we must first define it. Returning to the cookie example, each taste test yields an estimate of $\rho(\mathbf{x})$, the fraction of test subjects that like recipe \mathbf{x} . But uncertainty is always present, even after many measurements. Therefore, it is impossible to be certain that the constraint $\rho(\mathbf{x}) \geq 1 - \epsilon$ is satisfied for any \mathbf{x} . Likewise, the objective function can only be evaluated point-wise and, if noise is present, it may never be determined with certainty.

This is a stochastic programming problem: namely, an optimization problem in which the objective and/or constraints contain uncertain quantities whose probability distributions are known or can be estimated (see e.g., Shapiro et al., 2009). A natural formulation of these problems is to minimize the objective function *in expectation*, while satisfying the constraints *with high probability*. The condition that the constraint be satisfied with high probability is called a *probabilistic constraint*. This concept is formalized below.

Let $f(\mathbf{x})$ represent the objective function. Let $\mathcal{C}(\mathbf{x})$ represent the *constraint condition*, namely the boolean function indicating whether or not the constraint is satisfied for input \mathbf{x} . For example, in the cookie problem, $\mathcal{C}(\mathbf{x}) \iff \rho(\mathbf{x}) \geq 1 - \epsilon$. Then, our probabilistic constraint is

$$\Pr(\mathcal{C}(\mathbf{x})) \geq 1 - \delta, \quad (4)$$

for some user-specified minimum confidence $1 - \delta$.

If K constraints are present, for each constraint $k \in (1, \dots, K)$ we define $\mathcal{C}_k(\mathbf{x})$ to be the constraint condition for constraint k . Each constraint may also have its own tolerance δ_k , so we have K probabilistic constraints of the form

$$\Pr(\mathcal{C}_k(\mathbf{x})) \geq 1 - \delta_k. \quad (5)$$

All K probabilistic constraints must ultimately be satisfied at a solution to the optimization problem.¹

Given these definitions, a general class of constrained Bayesian optimization problems can be formulated as

$$\min_{\mathbf{x}} \mathbb{E}[f(\mathbf{x})] \text{ s.t. } \forall k \Pr(\mathcal{C}_k(\mathbf{x})) \geq 1 - \delta_k. \quad (6)$$

The remainder of this paper proposes methods for solving problems in this class using Bayesian optimization. Two key ingredients are needed: a model of the objective and constraint functions (Section 2), and an acquisition function that determines which input \mathbf{x} would be most beneficial to observe next (Section 3).

¹Note: this formulation is based on individual constraint satisfaction for all constraints. Another reasonable formulation requires the (joint) probability that *all* constraints are satisfied to be above some single threshold.

2 MODELING THE CONSTRAINTS

2.1 GAUSSIAN PROCESSES

We use Gaussian processes (GPs) to model both the objective function $f(\mathbf{x})$ and the constraint functions. A GP is a generalization of the multivariate normal distribution to arbitrary index sets, including infinite length vectors or functions, and is specified by its positive definite covariance kernel function $K(\mathbf{x}, \mathbf{x}')$. GPs allow us to condition on observed data and tractably compute the posterior distribution of the model for any finite number of query points. A consequence of this property is that the marginal distribution at any single point is univariate Gaussian with a known mean and variance. See Rasmussen and Williams (2006) for an in-depth treatment of GPs for machine learning.

We assume the objective and all constraints are independent and model them with independent GPs. Note that since the objective and constraints are all modeled independently, they need not all be modeled with GPs or even with the same types of models as each other. Any combination of models suffices, so long as each one represents its uncertainty about the true function values.

2.2 THE LATENT CONSTRAINT FUNCTION, $g(\mathbf{x})$

In order to model constraint conditions $\mathcal{C}_k(\mathbf{x})$, we introduce real-valued latent *constraint functions* $g_k(\mathbf{x})$ such that for each constraint k , the constraint condition $\mathcal{C}_k(\mathbf{x})$ is satisfied if and only if $g_k(\mathbf{x}) \geq 0$.² Different observation models lead to different likelihoods on $g(\mathbf{x})$, as discussed below. By computing the posterior distribution of $g_k(\mathbf{x})$ for each constraint, we can then compute $\Pr(\mathcal{C}_k(\mathbf{x})) = \Pr(g_k(\mathbf{x}) \geq 0)$ by simply evaluating the Gaussian CDF using the predictive marginal mean and variance of the GP at \mathbf{x} .

Different constraints require different definitions of the constraint function $g(\mathbf{x})$. When the nature of the problem permits constraint observations to be modeled with a Gaussian likelihood, the posterior distribution of $g(\mathbf{x})$ can be computed in closed form. If not, approximations or sampling methods are needed (see Rasmussen and Williams, 2006, p. 41-75). We discuss two examples below, one of each type, respectively.

2.3 EXAMPLE I: BOUNDED RUNNING TIME

Consider optimizing some property of a computer program such that its running time $\tau(\mathbf{x})$ must not exceed some

²Any inequality constraint $g(\mathbf{x}) \leq g_0$ or $g(\mathbf{x}) \geq g_1$ can be represented this way by transforming to a new variable $\hat{g}(\mathbf{x}) \equiv g_0 - g(\mathbf{x}) \geq 0$ or $\hat{g}(\mathbf{x}) \equiv g(\mathbf{x}) - g_1 \geq 0$, respectively, so we set the right-hand side to zero without loss of generality.

value τ_{\max} . Because $\tau(\mathbf{x})$ is a measure of time, it is non-negative for all \mathbf{x} and thus not well-modeled by a GP prior. We therefore choose to model time in logarithmic units. In particular, we define $g(\mathbf{x}) = \log \tau_{\max} - \log \tau$, so that the condition $g(\mathbf{x}) \geq 0$ corresponds to our constraint condition $\tau \leq \tau_{\max}$, and place a GP prior on $g(\mathbf{x})$. For every problem, this transformation implies a particular prior on the original variables; in this case, the implied prior on $\tau(\mathbf{x})$ is the log-normal distribution. In this problem we may also posit a Gaussian likelihood for observations of $g(\mathbf{x})$. This corresponds to the generative model that constraint observations are generated by some true latent function corrupted with i.i.d. Gaussian noise. As with the prior, this choice implies something about the original function $\tau(\mathbf{x})$, in this case a log-normal likelihood. The basis for these choices is their computational convenience. Given a Gaussian prior and likelihood, the posterior distribution is also Gaussian and can be computed in closed form using the standard GP predictive equations.

2.4 EXAMPLE II: MODELING COOKIE TASTINESS

Recall the cookie optimization, and let us assume that constraint observations arrive as a set of counts indicating the numbers of people who did and did not like the cookies. We call these *binomial constraint observations*. Because these observations are discrete, they are not modeled well by a GP prior. Instead, we model the (unknown) binomial probability $\rho(\mathbf{x})$ that a test subject likes cookie \mathbf{x} , which is linked to the observations through a binomial likelihood.³ In Section 1.5, we selected the constraint condition $\rho(\mathbf{x}) \geq 1 - \epsilon$, where $1 - \epsilon$ is the user-specified threshold representing the minimum allowable probability that a test subject likes the new cookie. Because $\rho(\mathbf{x}) \in (0, 1)$ and $g(\mathbf{x}) \in \mathbb{R}$, we define $g(\mathbf{x}) = s^{-1}(\rho(\mathbf{x}))$, where $s(\cdot)$ is a monotonically increasing sigmoid function mapping $\mathbb{R} \rightarrow (0, 1)$ as in logistic or probit regression.⁴ In our implementation, we use $s(z) = \Phi(z)$, the Gaussian CDF. The likelihood of $g(\mathbf{x})$ given the binomial observations is then the binomial likelihood composed with s^{-1} . Because this likelihood is non-Gaussian, the posterior distribution cannot be computed in closed form, and therefore approximation or sampling methods are needed.

2.5 INTEGRATING OUT THE GP HYPERPARAMETERS

Following Snoek et al. (2012), we use the Matérn 5/2 kernel for the Gaussian process prior, which corresponds to the

³We use the notation $\rho(\mathbf{x})$ both for the fraction of test subjects who like recipe \mathbf{x} and for its generative interpretation as the probability that a subject likes recipe \mathbf{x} .

⁴When the number of binomial trials is one, this model is called Gaussian Process Classification.

assumption that the function being modeled is twice differentiable. This kernel has $D + 1$ hyperparameters in D dimensions: one characteristic length scale per dimension, and an overall amplitude. Again following Snoek et al. (2012), we perform a fully-Bayesian treatment by integrating out these kernel hyperparameters with Markov chain Monte Carlo (MCMC) via slice sampling (Neal, 2000).

When the posterior distribution cannot be computed in closed form due to a non-Gaussian likelihood, we use elliptical slice sampling (Murray et al., 2010) to sample $g(\mathbf{x})$. We also use the prior whitening procedure described in Murray and Adams (2010) to avoid poor mixing due to the strong coupling of the latent values and the kernel hyperparameters.

3 ACQUISITION FUNCTIONS

3.1 CONSTRAINT WEIGHTED EXPECTED IMPROVEMENT

Given the probabilistic constraints and the model for a particular problem, it remains to specify an acquisition function that leads to efficient optimization. Here, we present an acquisition function for constrained Bayesian optimization under the Expected Improvement (EI) criterion (Section 1.2). However, the general framework presented here does not depend on this specific choice and can be used in conjunction with any improvement criterion.

Because improvement is not possible when the constraint is violated, we can define an acquisition function for constrained Bayesian optimization by extending the expectation in Eq. 1 to include the additional constraint uncertainty. This results in a constraint-weighted expected improvement criterion, $a(\mathbf{x})$:

$$a(\mathbf{x}) = \text{EI}(\mathbf{x}) \Pr(\mathcal{C}(\mathbf{x})) \quad (7)$$

$$= \text{EI}(\mathbf{x}) \prod_{k=1}^K \Pr(\mathcal{C}_k(\mathbf{x})) \quad (8)$$

where the second line follows from the assumed independence of the constraints. The gradient of this acquisition function is readily computed and therefore this acquisition function can be maximized in the same manner as standard EI. In practice we maximize it following the method in Snoek et al. (2012).

Then, the full acquisition function $a(\mathbf{x})$, after integrating out the GP hyperparameters, is given by

$$a(\mathbf{x}) = \int \text{EI}(\mathbf{x}|\theta)p(\theta|\mathbf{D})p(\mathcal{C}(\mathbf{x})|\mathbf{x}, \mathbf{D}', \omega)p(\omega|\mathbf{D}')d\theta d\omega,$$

where θ is the set of GP hyperparameters for the objective function model, ω is the set of GP hyperparameters for the constraint model(s), $\mathbf{D} = \{\mathbf{x}_n, y_n\}_{n=1}^N$ are the previous

objective function observations, and \mathbf{D}' are the constraint function observations.

3.2 FINDING THE FEASIBLE REGION

The acquisition function given above is not defined when at least one probabilistic constraint is violated for all \mathbf{x} , because in this case the EI target does not exist and therefore EI cannot be computed. In this case we take the acquisition function to include only the second factor,

$$a(\mathbf{x}) = \prod_{k=1}^K \Pr(g_k(\mathbf{x}) \geq 0) \quad (9)$$

Intuitively, if the probabilistic constraint is violated everywhere, we ignore the objective function and try to satisfy the probabilistic constraint until it is satisfied somewhere. This acquisition function may also be used if no objective function exists, i.e., if the problem is just to search for any feasible input. This feasibility search is purely exploitative: it searches where the probability of satisfying the constraints is highest. This is possible because the true probability of constraint satisfaction is either zero or one. Therefore, as the algorithm continues to probe a particular region, it will either discover that the region is feasible, or the probability will drop and it will move on to a more promising region.

3.3 DECOUPLED OBSERVATIONS

In some problems, the objective and constraint functions may be evaluated independently. We call this property the *decoupling* of the objective and constraint functions. In decoupled problems, we must choose to evaluate either the objective function or one of the constraint functions at each iteration of Bayesian optimization. As discussed in Section 1.3, it is important to identify problems with this decoupled structure, because often some of the functions are much more expensive to evaluate than others. Bayesian optimization with decoupled constraints is a form of multi-task Bayesian optimization (Swersky et al., 2013), in which the different black-boxes or *tasks* are the objective and decoupled constraint(s), represented by the set $\{\text{objective}, 1, 2, \dots, K\}$ for K constraints.

3.3.1 Chicken and Egg Pathology

One possible acquisition function for decoupled constraints is the expected improvement of individually evaluating each task. However, the myopic nature of the EI criterion causes a pathology in this formulation that prevents exploration of the design space. Consider a situation, with a single constraint, in which some feasible region has been identified and thus the current best input is defined, but a large unexplored region remains. Evaluating only the objective in this region could not cause improvement as our

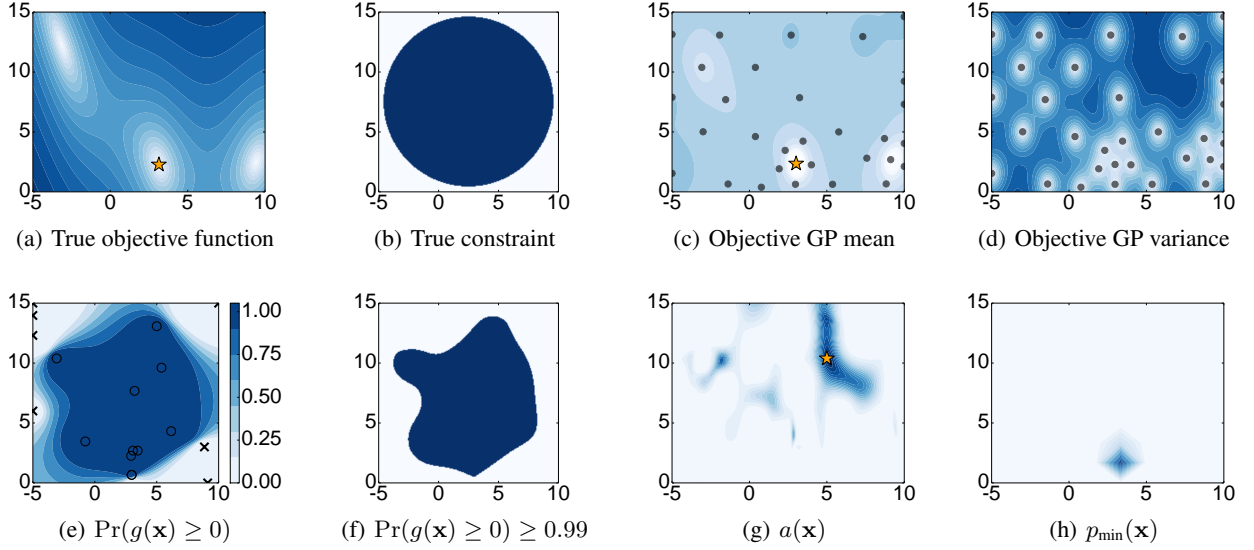


Figure 1: Constrained Bayesian optimization on the 2D Branin-Hoo function with a disk constraint, after 50 iterations (33 objective evaluations and 17 constraint evaluations): (a) Branin-Hoo function, (b) true constraint, (c) mean of objective function GP, (d) variance of objective function GP, (e) probability of constraint satisfaction, (f) probabilistic constraint, $\Pr(g(\mathbf{x}) \geq 0) \geq 0.99$, (g) acquisition function, $a(\mathbf{x})$, and (h) probability distribution over the location of the minimum, $p_{\min}(\mathbf{x})$. Lighter colors indicate lower values. Objective function observations are indicated with black circles in (c) and (d). Constraint observations are indicated with black \times 's (violations) and o 's (satisfactions) in (e). Orange stars: (a) unique true minimum of the constrained problem, (c) best solution found by Bayesian optimization, (g) input chosen for the next evaluation, in this case an objective evaluation because $\Delta S_o(\mathbf{x}) > \Delta S_c(\mathbf{x})$ at the next observation location \mathbf{x} .

belief about $\Pr(g(\mathbf{x}) \geq 0)$ will follow the prior and not exceed the threshold $1 - \delta$. Likewise, evaluating only the constraint would not cause improvement because our belief about the objective will follow the prior and is unlikely to become the new best. This is a causality dilemma: we must learn that *both* the objective and the constraint are favorable for improvement to occur, but this is not possible when only a single task is observed. This difficulty suggests a non-myopic acquisition function which assesses the improvement after a sequence of objective and constraint observations. However, such a multi-step acquisition function is intractable in general (Ginsbourger and Riche, 2010).

Instead, to address this pathology, we propose to use the coupled acquisition function (Eq. 7) to select an input \mathbf{x} for observation, followed by a second step to determine which task will be evaluated at \mathbf{x} . Following Swersky et al. (2013), we use the entropy search criterion (Hennig and Schuler, 2012) to select a task. However, our framework does not depend on this choice.

3.3.2 Entropy Search Criterion

Entropy search works by considering $p_{\min}(\mathbf{x})$, the probability distribution over the location of the minimum of the objective function. Here, we extend the definition of p_{\min} to be the probability distribution over the location of the solution to the constrained problem. Entropy search seeks

the action that, in expectation, most reduces the relative entropy between $p_{\min}(\mathbf{x})$ and an uninformative base distribution such as the uniform distribution. Intuitively speaking, we want to reduce our uncertainty about p_{\min} as much as possible at each step, or, in other words, maximize our information gain at each step. Following Hennig and Schuler (2012), we choose $b(\mathbf{x})$ to be the uniform distribution on the input space. Given this choice, the relative entropy of p_{\min} and b is the differential entropy of p_{\min} up to a constant that does not affect the choice of task. Our decision criterion is then

$$T^* = \arg \min_T \mathbb{E}_y \left[S(p_{\min}^{(y_T)}) - S(p_{\min}) \right], \quad (10)$$

where T is one of the tasks in $\{\text{objective}, 1, 2, \dots, K\}$, T^* is the selected task, $S(\cdot)$ is the differential entropy functional, and $p_{\min}^{(y_T)}$ is p_{\min} conditioned on observing the value y_T for task T . When integrating out the GP covariance hyperparameters, the full form is

$$T^* = \arg \min_T \int S(p_{\min}^{(y_T)}) p(y_T | \theta, \omega) dy_T d\theta d\omega \quad (11)$$

where y_T is a possible observed outcome of selecting task T and θ and ω are the objective and constraint GP hyperparameters respectively.⁵

⁵For brevity, we have omitted the base entropy term (which does not affect the decision T^*) and the explicit dependence of p_{\min} on θ and ω .

3.3.3 Entropy Search in Practice

Solving Eq. 11 poses several practical difficulties, which we address here in turn. First, estimating $p_{\min}(\mathbf{x})$ requires a discretization of the space. In the spirit of Hennig and Schuler (2012), we form a discretization of N_d points by taking the top N_d points according to the weighted expected improvement criterion. Second, p_{\min} cannot be computed in closed form and must be either estimated or approximated. Swersky et al. (2013) use Monte Carlo sampling to estimate p_{\min} by drawing samples from the GP on the discretization set and finding the minimum. We use the analogous method for constrained optimization: we sample from the objective function GP and all K constraint GPs, and then find the minimum of the objective for which the constraint is satisfied for all K constraint samples.

3.3.4 Incorporating cost information

Following Swersky et al. (2013), we incorporate information about the relative cost of the tasks by simply scaling the acquisition functions by these costs (provided by the user). In doing so, we pick the task with the most information gain per unit cost. If λ_A is the cost of observing task A , then Eq. 10 becomes

$$A^* = \arg \min_A \frac{1}{\lambda_A} \mathbb{E}_y \left[S \left(p_{\min}^{(y_A)} \right) - S(p_{\min}) \right]. \quad (12)$$

4 EXPERIMENTS

4.1 BRANIN-HOO FUNCTION

We first illustrate constrained Bayesian optimization on the Branin-Hoo function, a 2D function with three global minima (Fig. 1(a)). We add a decoupled disk constraint $(\mathbf{x}_1 - 2.5)^2 + (\mathbf{x}_2 - 7.5)^2 \leq 50$, shown in Fig. 1(b). This constraint eliminates the upper-left and lower-right solutions, leaving a unique global minimum at $\mathbf{x} = (\pi, 2.275)$, indicated by the orange star in Fig. 1(a). After 33 objective function evaluations and 17 constraint evaluations, the best solution is (3.01, 2.36), which satisfies the constraint and has value 0.48 (true best value = 0.40).

4.2 ONLINE LDA WITH SPARSE TOPICS

Online Latent Dirichlet Allocation (LDA, Hoffman et al., 2010) is an efficient variational formulation of a popular topic model for learning topics and corresponding word distributions given a corpus of documents. In order for topics to have meaningful semantic interpretations, it is desirable for the word distributions to exhibit sparsity. In this experiment we optimize the hyperparameters of online LDA subject to the constraint that the entropy of the per-topic word distribution averaged over topics is less than $\log_2 200$ bits, which is achieved, for example by allocating uniform density over 200 words. We used the online LDA

implementation from Agarwal et al. (2011) and optimized five hyperparameters corresponding to the number of topics (from 2 to 100), two Dirichlet distribution prior base measures (from 0 to 2), and two learning rate parameters (rate from 0.1 to 1, decay from 10^{-5} to 1). As a baseline, we compare with unconstrained Bayesian optimization in which constraint violations are set to the worst possible value for this LDA problem. Fig. 2(a) shows that constrained Bayesian optimization significantly outperforms the baseline and the IECI method from Gramacy and Lee (2010) (see Section 1.4). Intuitively, the baseline is poor because the GP has difficulty modeling the sharp discontinuities caused by the large values.

4.3 MEMORY-LIMITED NEURAL NET

In the final experiment, we optimize the hyperparameters of a deep neural network on the MNIST handwritten digit classification task in a memory-constrained scenario. We optimize over 11 parameters: 1 learning rate, 2 momentum parameters (initial and final), the number of hidden units per layer (2 layers), the maximum norm on model weights (for 3 sets of weights), and the dropout regularization probabilities (for the inputs and 2 hidden layers). We optimize the classification error on a withheld validation set under the constraint that the total number of model parameters (weights in the network) must be less than one million. This constraint is decoupled from the objective and inexpensive to evaluate, because the number of weights can be calculated directly from the parameters, without training the network. We train the neural network using momentum-based stochastic gradient descent which is notoriously difficult to tune as training can diverge under various combinations of the momentum and learning rate. When training diverges, the objective function cannot be measured. Reporting the constraint violation as a large objective value performs poorly because it introduces sharp discontinuities that are hard to model (Fig. 2). This necessitates a second noisy, binary constraint which is violated when training diverges, for example when the both the learning rate and momentum are too large. The network is trained⁶ for 25,000 weight updates and the objective is reported as classification error on the standard validation set. Our Bayesian optimization routine can thus choose between two decoupled tasks, evaluating the memory constraint or the validation error after a full training run. Evaluating the validation error can still cause a constraint violation when the training diverges, which is treated as a binary constraint in our model. Fig. 2(b) shows a comparison of our constrained Bayesian optimization against a baseline standard Bayesian optimization where constraint violations are treated as resulting in a random classifier (90% error). Only the objective evaluations are presented, since

⁶We use the Deepnet package: <https://github.com/nitishsrivastava/deepnet>.

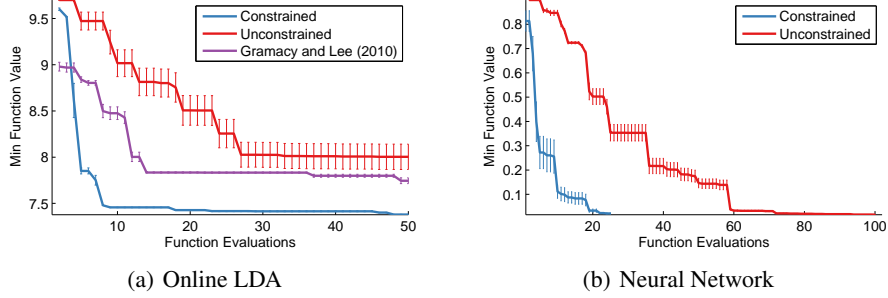


Figure 2: Empirical performance of constrained Bayesian optimization for (a) Online Latent Dirichlet Allocation and (b) turning a deep neural network. Blue curves: our method. Red curves: unconstrained Bayesian optimization with constraint violations as large values. Purple curve: Integrated Expected Conditional Improvement method from Gramacy and Lee (2010). Errors bars indicate standard error from 5 independent runs.

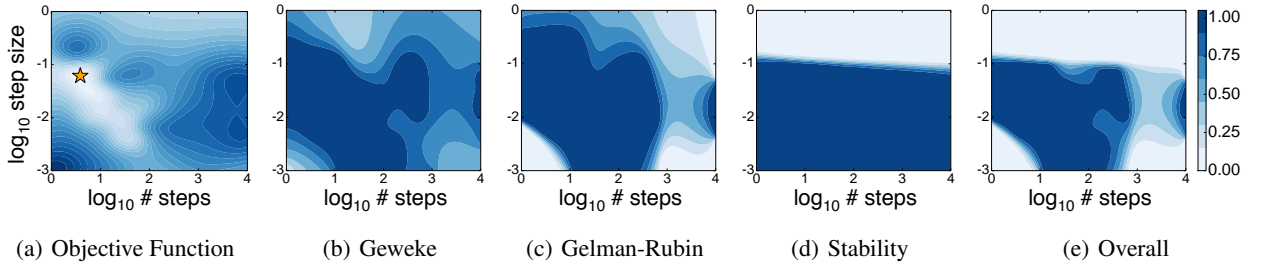


Figure 3: Tuning Hamiltonian Monte Carlo with constrained Bayesian optimization: (a) objective function model, (b-e) constraint satisfaction probability surfaces for (b) Geweke test, (c) Gelman-Rubin test, (d) stability of the numerical integration, (e) overall, which is the product of the preceding three probability surfaces. In (a), lighter colors correspond to more effective samples, circles indicate function evaluations, and the orange star indicates the best solution. Vertical axis label at left is for all subplots. Probability colormap at right is for (b-d).

constraint evaluations are extremely inexpensive compared to an entire training run. In the event that training diverges on an objective evaluation, we report 90% error. The optimized net has a learning rate of 0.1, dropout probabilities of 0.17 (inputs), 0.30 (first layer), and 0 (second layer), initial momentum 0.86, and final momentum 0.81. Interestingly, the optimization chooses a small first layer (size 312) and a large second layer (size 1772).

4.4 TUNING MCMC

Hamiltonian Monte Carlo (HMC) is a popular MCMC sampling technique that takes advantage of gradient information for rapid mixing. However, HMC contains several parameters that require careful tuning. The two basic parameters are the number of leapfrog steps τ , and the step size ϵ . HMC may also include a mass matrix which introduces $\mathcal{O}(D^2)$ additional parameters in D dimensions, although the matrix is often chosen to be diagonal (D parameters) or a multiple of the identity matrix (1 parameter) (Neal, 2011). In this experiment, we optimize the performance of HMC using Bayesian optimization; see Mahendran et al. (2012) for a similar approach. We optimize the following parameters: τ , ϵ , a mass parameter, and the frac-

tion of the allotted computation time spent burning in the chain.

Our experiment measures the number of effective samples (ES) in a fixed computation time; this corresponds to finding chains that minimize estimator variance. We impose the constraints that the generated samples must pass the Geweke (Geweke, 1992) and Gelman-Rubin (Gelman and Rubin, 1992) convergence diagnostics. In particular, we require the worst (largest absolute value) Geweke test score across all variables and chains to be at most 2.0, and the worst (largest) Gelman-Rubin score between chains and across all variables to be at most 1.2. We use PyMC (Patil et al., 2010) for the convergence diagnostics and the LaplacesDemon R package to compute effective sample size. The chosen thresholds for the convergence diagnostics are based on the PyMC and LaplacesDemon documentation. The HMC integration may also diverge for large values of ϵ ; we treat this as an additional constraint, and set $\delta = 0.05$ for all constraints. We optimize HMC sampling from the posterior of a logistic regression binary classification problem using the German credit data set from the UCI repository (Frank and Asuncion, 2010). The data set contains 1000 data points, and is normalized to have unit

Table 1: Tuning Hamiltonian Monte Carlo.

Experiment	burn-in	# steps, τ	step size, ϵ	mass	# samples	accept rate	effective samples
Baseline	10%	100	0.047	1	8.3×10^3	85%	1.1×10^3
BayesOpt	3.8%	2	0.048	1.55	3.3×10^5	70%	9.7×10^4

variance. We initialize each chain randomly with D independent draws from a Gaussian distribution with mean zero and standard deviation 10^{-3} . For each set of inputs, we compute two chains, each with 5 minutes of computation time on a single core of a compute node.

Fig. 3 shows constraint surfaces discovered by Bayesian optimization for a simpler experiment in which only τ and ϵ are varied; burn-in is fixed at 10% and the mass is fixed at 1. These diagrams yield interpretations of the feasible region; for example, Fig. 3(d) shows that the numerical integration diverges for values of ϵ above $\approx 10^{-1}$. Table 1 shows the results of our 4-parameter optimization after 50 iterations, compared with a baseline that is reflective of a typical HMC configuration: 10% burn in, 100 leapfrog steps, and the step size chosen to yield an 85% proposal accept rate. Each row in the table was produced by averaging 5 independent runs with the given parameters. The optimization chooses to perform very few ($\tau = 2$) leapfrog steps and spend relatively little time (3.8%) burning in the chain, and chooses an acceptance rate of 70%. In contrast, the baseline spends much more time generating each proposal ($\tau = 100$), which produces many fewer total samples and, correspondingly, significantly fewer effective samples.

5 CONCLUSION

In this paper we extended Bayesian optimization to constrained optimization problems. Because constraint observations may be noisy, we formulate the problem using probabilistic constraints, allowing the user to directly express the tradeoff between cost and risk by specifying the confidence parameter δ . We then propose an acquisition function to perform constrained Bayesian optimization, including the case where the objective and constraint(s) may be observed independently. We demonstrate the effectiveness of our system on the meta-optimization of machine learning algorithms and sampling techniques. Constrained optimization is a ubiquitous problem and we believe this work has applications in areas such as product design (e.g. designing a low-calorie cookie), machine learning meta-optimization (as in our experiments), real-time systems (such as a speech recognition system on a mobile device with speed, memory, and/or energy usage constraints), or any optimization problem in which the objective function and/or constraints are expensive to evaluate and possibly noisy.

Acknowledgements

The authors would like to thank Geoffrey Hinton, George Dahl, and Oren Rippel for helpful discussions, and Robert Nishihara for help with the experiments. This work was partially funded by DARPA Young Faculty Award N66001-12-1-4219. Jasper Snoek is a fellow in the Harvard Center for Research on Computation and Society.

References

- Alekh Agarwal, Olivier Chapelle, Miroslav Dudík, and John Langford. A reliable effective terascale linear learning system, 2011. arXiv: 1110.4198 [cs.LG].
- James S. Bergstra, Rémi Bardenet, Yoshua Bengio, and Bálázs Kégl. Algorithms for hyper-parameter optimization. In *NIPS*. 2011.
- Eric Brochu, Tyson Brochu, and Nando de Freitas. A Bayesian interactive optimization approach to procedural animation design. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2010a.
- Eric Brochu, Vlad M. Cora, and Nando de Freitas. A tutorial on Bayesian optimization of expensive cost functions, 2010b. arXiv:1012.2599 [cs.LG].
- Adam D. Bull. Convergence rates of efficient global optimization algorithms. *JMLR*, (3-4):2879–2904, 2011.
- Nando de Freitas, Alex Smola, and Masrour Zoghi. Exponential regret bounds for Gaussian process bandits with deterministic observations. In *ICML*, 2012.
- Josip Djolonga, Andreas Krause, and Volkan Cevher. High dimensional Gaussian Process bandits. In *NIPS*, 2013.
- Andrew Frank and Arthur Asuncion. UCI machine learning repository, 2010.
- Andrew Gelman and Donald R. Rubin. A single series from the Gibbs sampler provides a false sense of security. In *Bayesian Statistics*, pages 625–32. Oxford University Press, 1992.
- John Geweke. Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments. In *Bayesian Statistics*, pages 169–193. University Press, 1992.
- David Ginsbourger and Rodolphe Riche. Towards Gaussian process-based optimization with finite time horizon. In *Advances in Model-Oriented Design and Analysis*. Physica-Verlag HD, 2010.

- Robert B. Gramacy and Herbert K. H. Lee. Optimization under unknown constraints, 2010. arXiv:1004.4027 [stat.ME].
- Philipp Hennig and Christian J. Schuler. Entropy search for information-efficient global optimization. *JMLR*, 13, 2012.
- Matthew Hoffman, David M. Blei, and Francis Bach. Online learning for latent Dirichlet allocation. In *NIPS*, 2010.
- Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *LION*, 2011.
- Donald R. Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, 21, 2001.
- Andreas Krause and Cheng Soon Ong. Contextual Gaussian Process bandit optimization. In *NIPS*, 2011.
- Dan Lizotte. *Practical Bayesian Optimization*. PhD thesis, University of Alberta, Edmonton, Canada, 2008.
- Nimalan Mahendran, Ziyu Wang, Firas Hamze, and Nando de Freitas. Adaptive MCMC with Bayesian optimization. In *AISTATS*, 2012.
- Jonas Mockus, Vytautas Tiesis, and Antanas Zilinskas. The application of Bayesian methods for seeking the extremum. *Towards Global Optimization*, 2, 1978.
- Iain Murray and Ryan P. Adams. Slice sampling covariance hyperparameters of latent Gaussian models. In *NIPS*, 2010.
- Iain Murray, Ryan P. Adams, and David J.C. MacKay. Elliptical slice sampling. *JMLR*, 9:541–548, 2010.
- Radford Neal. Slice sampling. *Annals of Statistics*, 31: 705–767, 2000.
- Radford Neal. MCMC using Hamiltonian dynamics. In *Handbook of Markov Chain Monte Carlo*. Chapman and Hall/CRC, 2011.
- Anand Patil, David Huard, and Christopher Fonnesbeck. PyMC: Bayesian stochastic modelling in Python. *Journal of Statistical Software*, 2010.
- Carl Rasmussen and Christopher Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.
- A. Shapiro, D. Dentcheva, and A. Ruszczyński. *Lectures on stochastic programming: modeling and theory*. MPS-SIAM Series on Optimization, Philadelphia, USA, 2009.
- Jasper Snoek. *Bayesian Optimization and Semiparametric Models with Applications to Assistive Technology*. PhD thesis, University of Toronto, Toronto, Canada, 2013.
- Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical Bayesian optimization of machine learning algorithms. In *NIPS*, 2012.
- Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: no regret and experimental design. In *ICML*, 2010.
- Kevin Swersky, Jasper Snoek, and Ryan P. Adams. Multi-task Bayesian optimization. In *NIPS*, 2013.
- Ziyu Wang, Masrour Zoghi, Frank Hutter, David Matheson, and Nando de Freitas. Bayesian optimization in high dimensions via random embeddings. In *IJCAI*, 2013.
- Marcela Zuluaga, Andreas Krause, Guillaume Sergent, and Markus Püschel. Active learning for multi-objective optimization. In *ICML*, 2013.

Nonparametric Clustering with Distance Dependent Hierarchies

Soumya Ghosh

Dept. of Computer Science,
Brown Univ., Providence, RI
sghosh@cs.brown.edu

Michalis Raptis

Comcast Labs,
Washington, D.C.
mraptis@cable.comcast.com

Leonid Sigal

Disney Research,
Pittsburgh, PA
lsigal@disneyresearch.com

Erik B. Sudderth

Dept. of Computer Science,
Brown Univ., Providence, RI
sudderth@cs.brown.edu

Abstract

The distance dependent Chinese restaurant process (ddCRP) provides a flexible framework for clustering data with temporal, spatial, or other structured dependencies. Here we model multiple groups of structured data, such as pixels within frames of a video sequence, or paragraphs within documents from a text corpus. We propose a hierarchical generalization of the ddCRP which clusters data within groups based on distances between data items, and couples clusters across groups via distances based on aggregate properties of these local clusters. Our hddCRP model subsumes previously proposed hierarchical extensions to the ddCRP, and allows more flexibility in modeling complex data. This flexibility poses a challenging inference problem, and we derive a MCMC method that makes coordinated changes to data assignments both within and between local clusters. We demonstrate the effectiveness of our hddCRP on video segmentation and discourse modeling tasks, achieving results competitive with state-of-the-art methods.

1 INTRODUCTION

The recent explosive growth of image and video repositories, and of structured data collections more broadly, motivates methods for the unsupervised discovery of informative latent structures. Image sequences of course exhibit strong spatio-temporal dependencies: objects typically occupy blocks of spatially contiguous pixels, and their movements induce strong dependencies among video frames. Nevertheless, many previous nonparametric models for visual data have mostly ignored such relationships, relying on careful feature engineering to make local likelihoods informative (Sudderth et al., 2008; Haines & Xiang, 2012). While accounting for spatial dependencies can be technically challenging, it produces image partitions which much more accurately reflect real-world scene structure (Orbanz

& Buhmann, 2008; Sudderth & Jordan, 2008). However, these methods treat images as an unordered, or *exchangeable*, collection; they thus fail to capture the strong temporal dependencies found in video sequences.

Blei & Frazier (2011) proposed the *distance dependent Chinese restaurant process* (ddCRP) as a flexible distribution over partitions of data with temporal, spatial, or other non-exchangeable structure. The ddCRP represents partitions via links between data instances: each observation links to one other, and the probability of linking to nearby instances is higher. Closeness is measured according to a distance which may be arbitrarily specified to capture domain knowledge. The connected components of the induced link graph then partition the dataset into clusters. Previous work has used the ddCRP to effectively cluster data with sequential, temporal, or spatial structure (Ghosh et al., 2011; Socher et al., 2011; Ghosh et al., 2012).

In this paper, we propose a *hierarchical ddCRP* (hddCRP) that captures local relationships like these, but also uses distances among latent clusters to extract further global dependencies. After an initial ddCRP partitioning, local clusters are grouped via additional links that depend on a user-specified measure of cluster similarity. This framework allows the hddCRP to model relationships that depend on *aggregate* properties of clusters such as size and shape, which may be difficult to capture with likelihoods alone. Given arbitrary cluster and data affinity functions, which need not arise from true distance metrics, the hddCRP always defines a valid joint probability distribution on partitions.

The hddCRP is a hierarchical generalization of the ddCRP which unifies and generalizes existing models. Simpler hierarchical extensions of the ddCRP employing restricted distance functions (Ghosh et al., 2011; Kim & Oh, 2011), as well as the “Chinese restaurant franchise” representation of the *hierarchical Dirichlet process* (HDP, Teh et al. (2006)), are special cases of the hddCRP. The HDP and related dependent Dirichlet process models (MacEachern, 1999) define dependent random measures from which allocations of data to clusters are sampled, indirectly inducing dependencies in the resulting partitions. For example,

Griffin & Steel (2006), Dunson & Park (2008), Rao & Teh (2009), and Lin et al. (2010) define priors which encourage “close” data points to have similar allocation distributions.

In contrast, the hddCRP directly specifies distributions over partitions via a flexible set of user-specified affinity functions. This allows structural constraints on clusters, such as connectivity (Ghosh et al., 2011), to be directly enforced. The hddCRP does not require its “distance” functions to be true metrics or have any special properties, and thus provides an extremely flexible framework for modeling complex data. Alternative models based on latent Gaussian processes (Duan et al., 2007; Sudderth & Jordan, 2008) require appropriate positive-definite kernel functions, whose specification can be challenging in non-Euclidean spaces (e.g., of object shapes). By working directly with discrete partitions, rather than latent continuous measures, the hddCRP also allows more computationally efficient inference.

The hddCRP generative process defined in Section 2 is simple, but the data-level and cluster-level link variables are strongly coupled in the posterior. Section 3 develops a *Markov chain Monte Carlo* (MCMC) method that makes coordinated changes to links at both levels, and thus more effectively explores clustering hypotheses. This sampler is also a novel inference algorithm for the HDP that makes large changes to the partition structure, without needing to explicitly craft split or merge proposals (Jain & Neal, 2004; Wang & Blei, 2012). By reasoning about data and cluster links, our sampler changes cluster allocations at varying resolutions, perturbing both memberships of data instances to local clusters and clusters to global components.

In Section 4, we demonstrate the versatility of the hddCRP by applying it to the problems of video segmentation and discourse analysis. In addition to having diverse data types (video sequences versus text documents), these two problems exhibit very different kinds of relationships among data instances and latent clusters. Nevertheless, our hddCRP model and inference framework easily applies to both domains by selecting appropriate data and cluster-level affinity functions. In both domains, explicit modeling of dependencies between latent clusters boosts performance over models that ignore such relationships.

2 HIERARCHICAL DISTANCE DEPENDENT CLUSTERS

The distance-dependent CRP (Blei & Frazier, 2011) defines a distribution over partitions indirectly via distributions over links between data instances. A data point i has an associated link variable c_i which links to another data instance j , or itself, according to the following distribution:

$$p(c_i = j \mid A, \alpha) \propto \begin{cases} A_{ij} & i \neq j, \\ \alpha & i = j. \end{cases} \quad (1)$$

The *affinity* $A_{ij} = f(d(i, j))$ depends on a user-specified *distance* $d(i, j)$ between pairs of data points, and a mono-

tonically decreasing *decay function* $f(d)$ which makes links to nearby data more likely. The resulting link structure induces a partition, where two data instances are assigned to the same cluster if and only if one is reachable from the other by traversing the link edges. Larger self-affinity parameters α favor partitions with more clusters.

2.1 THE HIERARCHICAL ddCRP

We propose a novel generative model that applies the ddCRP formalism twice, first for clustering data within each group into local clusters, and then for coupling the local clusters across groups. Like the ddCRP, our hddCRP defines a valid distribution over partitions of a dataset. It places higher probability mass on partitions that group nearby data points into latent clusters, *and* couple similar local clusters into global components. Examples of these data and cluster links are illustrated in Figure 1.

Consider a collection of G groups, where group g contains N_g observations. We denote the i^{th} data point of group g by x_{gi} , and the full dataset by \mathbf{x} . The data link variable c_{gi} for x_{gi} is sampled from a group-specific ddCRP:

$$p(c_{gi} = gj \mid \alpha_g, A^g) \propto \begin{cases} A_{ij}^g & i \neq j, \\ \alpha_g & i = j. \end{cases} \quad (2)$$

At this first level of link variables, we set the probability of linking observations in different groups to zero. The connected components of the links $c_g = \{c_{gi} \mid i = 1, \dots, N_g\}$ then determine the local clustering for group g .

Data links $\mathbf{c} = \{c_1, \dots, c_G\}$ across all groups divide the dataset into group-specific local clusters $T(\mathbf{c})$. The hddCRP then associates each cluster $t \in T(\mathbf{c})$ with a cluster link k_t drawn from a global ddCRP distribution:

$$p(k_t = s \mid \alpha_0, A^0(\mathbf{c})) \propto \begin{cases} A_{ts}^0(\mathbf{c}) & t \neq s, \\ \alpha_0 & t = s. \end{cases} \quad (3)$$

Here α_0 is a global self-affinity parameter, and $A^0(\mathbf{c})$ is the set of pairwise affinities between the elements of $T(\mathbf{c})$. We let $A_{ts}^0(\mathbf{c}) = f_0(d_0(t, s, \mathbf{c}))$, where $d_0(t, s, \mathbf{c})$ is a “distance” based on arbitrary properties of clusters t and s , and $f_0(d_0)$ a decreasing decay function. The connected components of $\mathbf{k} = \{k_t \mid t \in T(\mathbf{c})\}$ then couple local clusters into global components shared across groups. Let z_{gi} denote the component associated with observation i in group g , and $\mathbf{z} = \{z_{gi} \mid g = 1, \dots, G; i = 1, \dots, N_g\}$. Data instances x_{gi} and x_{hj} are clustered ($z_{gi} = z_{hj}$) if and only if they are reachable via some combination of data and cluster links.

Given this partition structure, we endow component m with likelihood parameters $\phi_m \sim G_0(\lambda)$, and generate observations $x_{gi} \sim p(x_{gi} \mid \phi_{z_{gi}})$. Let $M(\mathbf{c}, \mathbf{k})$ equal the number of global components induced by the cluster links \mathbf{k} and data links \mathbf{c} . Because data links \mathbf{c} are conditionally independent given $A^{1:G}$, and cluster links \mathbf{k} are conditionally independent given \mathbf{c} and the cluster affinities $A^0(\mathbf{c})$, the hddCRP

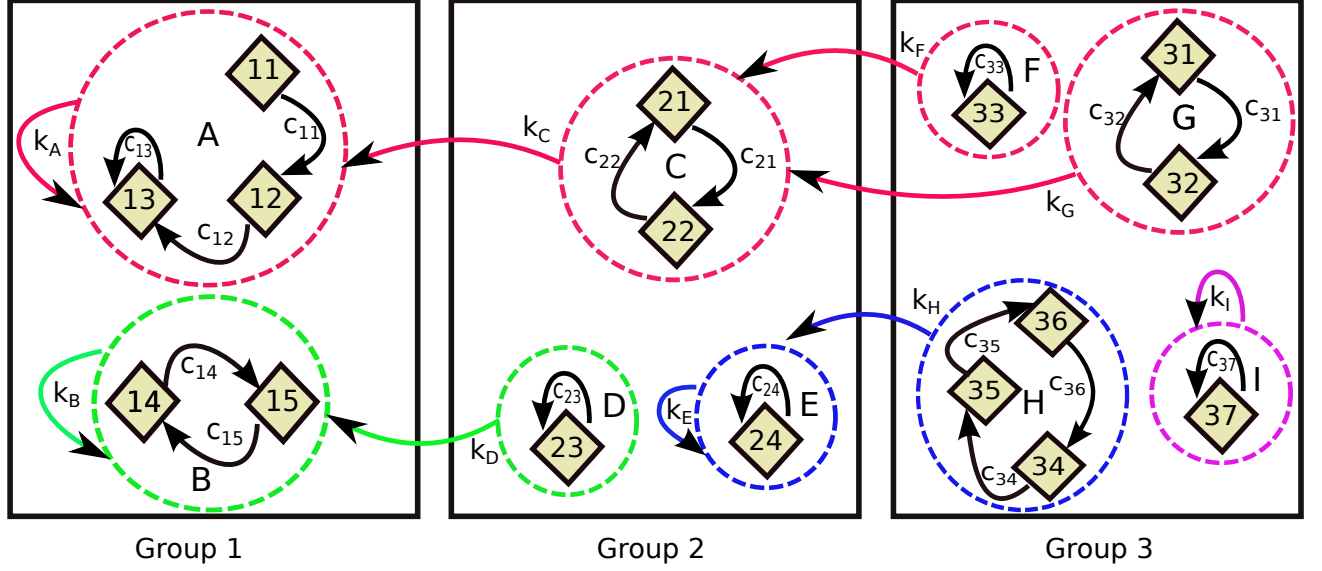


Figure 1: An example link variable configuration for a hierarchical ddCRP model of three groups (rectangles). Observed data points (customers, depicted as diamonds) link to other data points in the same group (black arrows), producing local clusters (dashed circles, labeled A through I). Cluster links (colored arrows) then join clusters to produce (in this case, four) global mixture components.

joint distribution on partitions and observations equals

$$p(\mathbf{x}, \mathbf{k}, \mathbf{c} \mid \alpha_{1:G}, \alpha_0, A^{1:G}, A^0, \lambda) = \prod_{m=1}^{M(\mathbf{c}, \mathbf{k})} p(x_{\mathbf{z}=m} \mid \lambda) \prod_{g=1}^G \prod_{i=1}^{N_g} p(c_{gi} \mid \alpha_g, A^g) \prod_{k_t \in \mathbf{k}} p(k_t \mid \mathbf{c}, \alpha_0, A^0(\mathbf{c})) \quad (4)$$

The set of data in component m is denoted by $x_{\mathbf{z}=m}$, and

$$p(x_{\mathbf{z}=m} \mid \lambda) = \int \prod_{gi|z_{gi}=m} p(x_{gi} \mid \phi_m) dG_0(\phi_m \mid \lambda), \quad (5)$$

where λ are hyperparameters specifying the prior distribution G_0 . Our inference algorithms assume this integral is tractable, as it always is when an exponential family likelihood is coupled with an appropriate conjugate prior. We emphasize that for arbitrary data and cluster affinities, the sequential hddCRP generative process defines a valid joint distribution $p(\mathbf{x}, \mathbf{k}, \mathbf{c}) = p(\mathbf{c})p(\mathbf{k} \mid \mathbf{c})p(\mathbf{x} \mid \mathbf{k}, \mathbf{c})$.

2.2 RELATED HIERARCHICAL MODELS

The hddCRP subsumes several recently proposed hierarchical extensions to the ddCRP, as well as the HDP itself, by defining appropriately restricted data affinities and local cluster affinities. Blei & Frazier (2011) show that the CRP is recovered from the ddCRP by arranging data in an arbitrary sequential order, and defining affinities as

$$A_{ij} = \begin{cases} 1 & \text{if } i < j, \\ 0 & \text{if } i > j. \end{cases} \quad (6)$$

Data points link to all previous observations with equal probability, and thus the probability of joining any existing cluster is proportional to the number of other data points already in that cluster. The probability of creating a new

cluster is proportional to the self-connection weight α . The resulting distribution on partitions can be shown to be invariant to the chosen sequential ordering of the data, and thus the standard CRP is *exchangeable* (Pitman, 2002).

Hierarchical Chinese Restaurant Process (hCRP) The hCRP representation of the HDP, which Teh et al. (2006) call the “Chinese restaurant franchise”, is recovered from the hddCRP by first defining group-specific affinities as in Eq. (6). We then arrange local clusters (tables, in the CRF metaphor) t sequentially with distances $A_{ts}^0(\mathbf{c}) = 1$ if $t < s$, and $A_{ts}^0(\mathbf{c}) = 0$ if $t > s$. Just as the two-level hCRP arises from a sequence of CRPs, the hddCRP is defined from a sequence of two ddCRP models.

Naive Hierarchical ddCRP (naive-hddCRP) The image segmentation model of Ghosh et al. (2011) clusters data within each group via a ddCRP based on an informative distance (in their experiments, spatial distance between image pixels). A standard CRP, as in the upper level of the HDP, is then used to combine these clusters into larger segments. Inference is substantially simpler for this special case, because cluster distances do not depend on properties of the data assigned to those clusters.

Distance Dependent Chinese Restaurant Franchise An alternate approach to capturing group-specific metadata uses a standard CRP to locally cluster data, but then uses the group labels to define affinities between clusters. Kim & Oh (2011) use this model to learn topic models of time-stamped documents. By constraining cluster affinities to depend on group labels, but not properties of the data assigned to within-group clusters, inference is simplified.

3 MCMC INFERENCE

The posterior distribution over the data and cluster links $p(\mathbf{c}, \mathbf{k} \mid \mathbf{x}, \alpha_{1:G}, \alpha_0, A^{1:G}, A^0, \lambda)$ is intractable, and we thus explore it via a Metropolis-Hastings MCMC method. Our approach generalizes the non-hierarchical ddCRP Gibbs sampler of Blei & Frazier (2011), which iteratively samples single data links conditioned on the observations and other data links. Evolving links lead to splits, merges, and other large changes to the partition structure. In the hddCRP, local clusters belong to global components, and these component memberships must be sampled as well.

3.1 MARKOV CHAIN STATE SPACE

The number of possible non-empty subsets (clusters) of N data points is $2^N - 1$. The state space of our Markov chain consists of the data links \mathbf{c} , and the set of *all* possible cluster links \mathcal{K} , one for each candidate non-empty cluster. For instance, given three observations $\{h, i, j\}$ the set of non-empty subsets is $\mathcal{T} = \{\{h\}, \{i\}, \{j\}, \{hi\}, \{ij\}, \{jh\}, \{hij\}\}$, and the corresponding set of possible cluster links is $\mathcal{K} = \{k_h, k_i, k_j, k_{hi}, k_{ij}, k_{jh}, k_{hij}\}$, where $|\mathcal{K}| = 2^3 - 1$.

For any configuration of \mathbf{c} , a strict subset of \mathcal{T} will have data associated with it. We call this the *active set*. For instance, if $c_h = h, c_i = i, c_j = j$, then only the clusters $\{\{h\}, \{i\}, \{j\}\}$ and the corresponding links $\{k_h, k_i, k_j\}$ are active. Given \mathbf{c} , we split \mathcal{K} into the active set \mathbf{k} , and the remaining inactive cluster links $\tilde{\mathbf{k}} = \mathcal{K} \setminus \mathbf{k}$. We account for the inactive clusters by augmenting $A^0(\mathbf{c})$ as follows:

$$\tilde{A}^0(\mathbf{c}) = \begin{bmatrix} A^0(\mathbf{c}) & \mathbf{0} \\ \mathbf{0} & \alpha_0 \mathbf{I} \end{bmatrix}. \quad (7)$$

Here, we have sorted the links so that affinities among the active clusters are listed in the upper-left quadrant of $\tilde{A}^0(\mathbf{c})$. As indicated by the identity matrix \mathbf{I} , inactive clusters have zero affinity with all other clusters, and link to themselves with probability one. Under this augmented model, the joint probability factorizes as follows:

$$\begin{aligned} p(\mathbf{x}, \mathbf{k}, \tilde{\mathbf{k}}, \mathbf{c}) &= p(\mathbf{c})p(\mathbf{k} \mid \mathbf{c})p(\tilde{\mathbf{k}} \mid \mathbf{c})p(\mathbf{x} \mid \mathbf{c}, \mathbf{k}, \tilde{\mathbf{k}}) = \\ &= p(\mathbf{c})p(\mathbf{k} \mid \mathbf{c})p(\tilde{\mathbf{k}} \mid \mathbf{c})p(\mathbf{x} \mid \mathbf{c}, \mathbf{k}) = p(\mathbf{x}, \mathbf{k}, \mathbf{c})p(\tilde{\mathbf{k}} \mid \mathbf{c}). \end{aligned} \quad (8)$$

Here, we have recovered the joint distribution of Eq. (4) because given \mathbf{c} , the observations \mathbf{x} are conditionally independent of the inactive links $\tilde{\mathbf{k}}$. Crucially, because inactive cluster links have no uncertainty, we must only explicitly represent the active clusters at each MCMC iteration.

As the Markov chain evolves, clusters are swapped in and out of the active set. Although the number of active clusters varies with the state of the chain, the dimensionality of the augmented state space $(\mathbf{c}, \mathbf{k}, \tilde{\mathbf{k}})$ remains constant, allowing us to ignore complications that arise when dealing with chains whose state spaces have varying dimensionality. In particular, we employ standard *Metropolis-Hastings* (MH) proposals to change data and cluster links, and need not resort to reversible jump MCMC (Green, 1995).

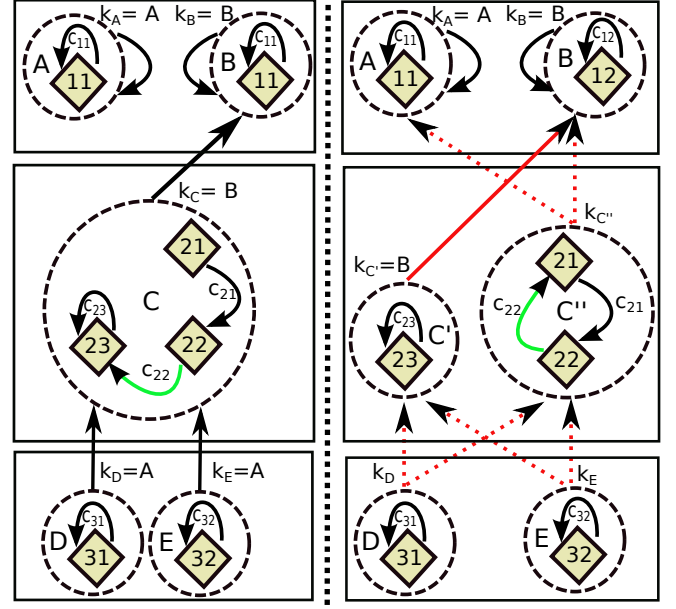


Figure 2: Illustration of changes induced by a data link proposal. Changing c_{22} (in the left configuration) splits cluster C into two clusters C' and C'' . The cluster links associated with C (shown in red) must also be resampled. The MH step of the sampler proposes a joint configuration of the links $\{c_{22}, k_{C'}, k_{C''}, k_D, k_E\}$. The dashed red arrows illustrate the possible values the resampled cluster links could take. A single data link can create large changes to the partition structure, with local clusters splitting or merging, and groups of clusters shifting between components.

3.2 LINK PROPOSAL DISTRIBUTIONS

In samplers previously developed for the hCRP (Teh et al., 2006) and the naive-hddCRP (Ghosh et al., 2011), local clusters directly sample their global component memberships. However for the hddCRP, cluster links indirectly determine global component memberships. This complicates inference, as any change to the cluster structure necessitates coordinated changes to the cluster links. As illustrated in Figure 2, consider the case where a data link proposal causes a cluster to break into two components. The new cluster must sample a cluster (outgoing) link, and cluster links pointing to the old cluster (incoming links) must be divided among the newly split clusters. Thus, we use a MH proposal to jointly resample data and affected cluster links.

To simplify the exposition, we focus on a particular group g and denote c_{gi} as c_i . Let the current state of the sampler be $\mathbf{k}(\mathbf{c})$ and $\mathbf{c} = \{c_{-i}, c_i = j\}$, so that i and j are members of the same cluster t_{ij} . Let $\mathcal{K}_{t_{ij}} = \{k_s \mid k_s = t_{ij}, s \neq t_{ij}\}$ denote the set of other clusters linking to t_{ij} .

Split? To construct our link proposal, we first set $c_i = i$. This may split current cluster t_{ij} into two new clusters, in which case we let t_i denote the cluster containing data i , and t_j the cluster containing formerly linked data j . Or, the partition structure may be unchanged so that $t_i = t_{ij}$.

Incoming links $k_s \in \mathcal{K}_{t_{ij}}$ to a split cluster are independently assigned to the new clusters with equal probability:

$$q_{\text{in}}(\mathcal{K}_{t_{ij}}) = \prod_{k_s \in \mathcal{K}_{t_{ij}}} \left(\frac{1}{2}\right)^{\delta(k_s, t_i)} \left(\frac{1}{2}\right)^{\delta(k_s, t_j)}. \quad (9)$$

The current outgoing link is retained by one of the split clusters, $k_{t_j} = k_{t_{ij}}$. To allow likelihood-based link proposals, we *temporarily* fix the other cluster link as $k_{t_i} = t_i$.

Propose Link We compare two proposals for c_i , the ddCRP prior distribution $q(c_i) = p(c_i | \alpha, A)$, and a data-dependent “pseudo-Gibbs” proposal distribution:

$$\begin{aligned} q(c_i) &\propto p(c_i | \alpha, A) \Gamma(\mathbf{x}, \mathbf{z}(c_i, \mathbf{c}_{-i}, \mathbf{k})), \\ &\Gamma(\mathbf{x}, \mathbf{z}(c_i, \mathbf{c}_{-i}, \mathbf{k})) \\ &= \begin{cases} \frac{p(\mathbf{x}_{\mathbf{z}=m_a} \cup \mathbf{x}_{\mathbf{z}=m_b} | \lambda)}{p(\mathbf{x}_{\mathbf{z}=m_a} | \lambda)p(\mathbf{x}_{\mathbf{z}=m_b} | \lambda)} & \text{if } c_i \text{ merges } m_a, m_b, \\ 1 & \text{otherwise.} \end{cases} \end{aligned} \quad (10)$$

The prior proposal, although naïve, can perform reasonably when A is sparse. The pseudo-Gibbs proposal is more sophisticated, as data links are proposed conditioned on both the observations \mathbf{x} and the current state of the sampler. Our experiments in Sec. 4 show it is much more effective.

Merge? Let $c_i = j^*$ denote the new data link sampled according to either the ddCRP prior or Eq. (10). Relative to the reference configuration in which $c_i = i$, this link may either leave the partition structure unchanged, or cause clusters t_i and t_{j^*} to merge into t_{ij^*} . In case of a merge, the new cluster retains the current outgoing link $k_{t_{ij^*}} = k_{t_{j^*}}$, and inherits the incoming links $\mathcal{K}_{t_{ij^*}} = \mathcal{K}_{t_i} \cup \mathcal{K}_{t_{j^*}}$.

If a merge does not occur, but t_{ij} was previously split into t_i and t_j , the outgoing link $k_{t_j} = k_{t_{ij}}$ is kept fixed. For newly created cluster t_i , we then propose a corresponding cluster link k_{t_i} from its full conditional distribution:

$$q_{\text{out}}(k_{t_i}) = p(k_{t_i} | \alpha_0, A^0(\mathbf{c}), \mathbf{x}, \mathbf{k}_{-t_i}, \mathbf{c}). \quad (11)$$

Note that the proposal $c_i = j^*$ may leave the original partition unchanged if $c_i = i$ does not cause t_{ij} to split, and $c_i = j^*$ does not result in a merge. In this case, the corresponding cluster links are also left unchanged.

Accept or Reject Combining the two pairs of cases above, our overall proposal distribution equals

$$q(\mathbf{c}^*, \mathbf{k}^* | \mathbf{c}, \mathbf{k}, \mathbf{x}) = \begin{cases} q(c_i^*) q_{\text{in}}(\mathcal{K}_{t_{ij}}^*) & \text{split, merge,} \\ q(c_i^*) & \text{no split, merge,} \\ q(c_i^*) q_{\text{out}}(k_{t_i}^*) q_{\text{in}}(\mathcal{K}_{t_{ij}}^*) & \text{split, no merge,} \\ p(c_i^* | \alpha, A) & \text{otherwise.} \end{cases}$$

Here, \mathbf{c}^* and \mathbf{k}^* denote the proposed values, which are then accepted or rejected according to the MH rule. For acceptance ratio derivations and further details, please see the supplemental material. After cycling through all data links \mathbf{c} , we use the Gibbs update of Eq. (11) to resample the cluster links \mathbf{k} , analogously to a standard ddCRP.

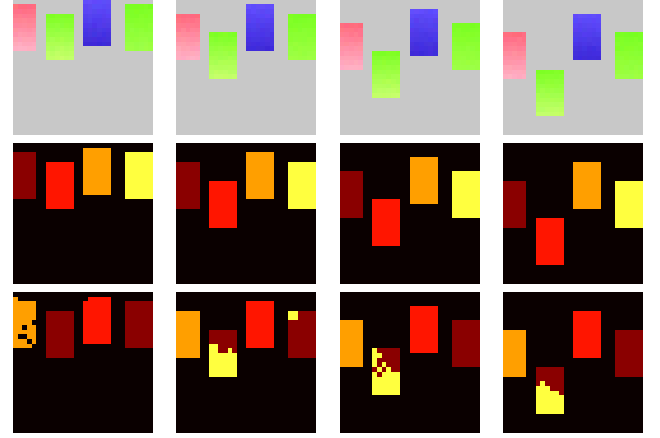


Figure 3: Experiments on synthetic data. *Top*: Ground truth partitions of a toy dataset containing four groups. Each group contains four objects exhibiting motion and color gradients. *Middle*: MAP partitions inferred by an hddCRP using size and optical flow-based cluster affinities. *Bottom*: MAP partitions discovered by a baseline hCRP using only color-based likelihoods.

4 EXPERIMENTS

In this section we present a series of experiments investigating the properties of the hddCRP model and our proposed MCMC inference algorithms. We examine a pair of challenging real-world tasks, video and discourse segmentation. We quantify performance by measuring agreement with held-out human annotations via the Rand index (Rand, 1971) and the WindowDiff metric (Pevzner & Hearst, 2002), demonstrating competitive performance.

To provide intuition, we first compare the hddCRP with the hCRP on a synthetic dataset (Figure 3) with four 30×30 frames (groups). Each frame contains four objects moving from top to bottom at different rates, and object appearances exhibit small color gradients. The hddCRP utilizes data link affinities that allow pixels (data instances) to connect to one of their eight spatial neighbors with equal probability. To exploit the differing motions of the objects, we define optical flow-based cluster affinities (Sun et al., 2010). Letting $w(t)$ denote the spatial positions occupied by cluster t after being warped by optical flow, and $w(s)$ the corresponding support of cluster s , the affinity is defined as $A_{ts}^0(\mathbf{c}) = (w(t) \cap w(s)) / (w(t) \cup w(s))$, $t \neq s$, encouraging clusters to link to other clusters with similar spatial support. Given this affinity function, hddCRP was able to robustly disambiguate the four uniquely moving objects, while the hCRP produced noisy segmentations and consistently confused objects with local similarity but distinct motion.

4.1 VIDEO SEGMENTATION

Likelihood As a preprocessing step, we divide each frame into approximately 1200 superpixels using the

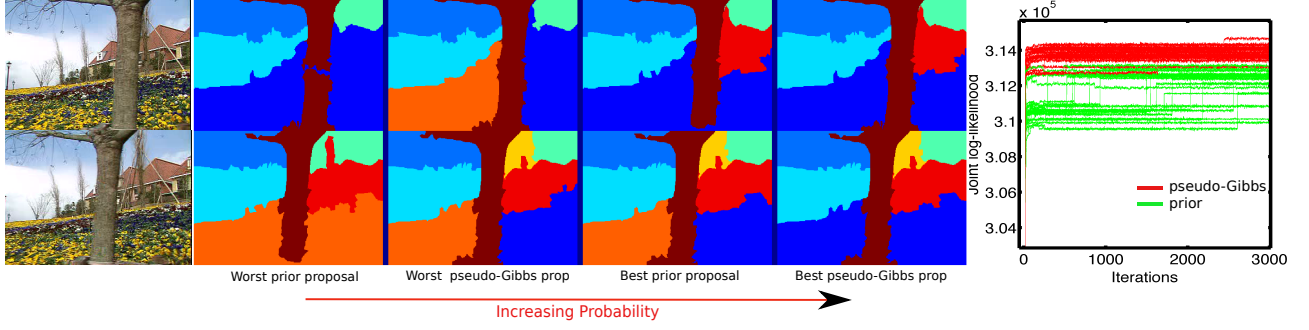


Figure 4: Data link proposal comparisons. *Left*: Two frames from the “garden” sequence, and partitions corresponding to the best and worst MAP samples using prior or pseudo-Gibbs proposals. *Right*: Joint log-likelihood trace plots for 25 trials of each proposal.

method proposed by Chang et al. (2013).¹ Each super-pixel is described by L_2 unit-normalized 120-bin HSV color and 128-bin local texton histograms. Unit normalization projects raw histograms to the surface of a hypersphere, where we use *von-Mises Fisher* (vMF) distributions (Mardia & Jupp, 2009) shared across all clusters of a global component. In preliminary experiments, we found that the vMF produced more accurate segmentations than multinomial models of raw histograms; similar L_2 normalizations are useful for image retrieval (Arandjelović & Zisserman, 2012). We also extracted optical flow using the “Classic+NL” algorithm (Sun et al., 2010), and associated a two-dimensional flow vector to each super-pixel, the median flow of its constituent pixels.

The color, texture, and flow features for super-pixel i in video frame g are denoted by $x_{gi} = \{x_{gi}^c, x_{gi}^t, x_{gi}^f\}$, where

$$x_{gi}^c \sim \text{vMF}(\mu_{z_{gi}}^c, \kappa^c), \mu_{z_{gi}}^c \sim \text{vMF}(\mu_0^c, \kappa_0^c), \quad (12)$$

where κ^c , μ_0^c , and κ_0^c are hyper-parameters controlling the concentration of color features around the direction $\mu_{z_{gi}}^c$, the mean color direction μ_0^c , and the concentration of $\mu_{z_{gi}}^c$ around μ_0^c . Texture features are generated similarly. Flow features are modeled via Gaussian distributions with conjugate, normal-inverse-Wishart priors:

$$\begin{aligned} x_{gi}^f &\sim \mathcal{N}(\mu_{z_{gi}}^f, \Sigma_{z_{gi}}^f), \Sigma_{z_{gi}}^f \sim \mathcal{IW}(n_0, S_0), \\ \mu_{z_{gi}}^f &| \Sigma_{z_{gi}}^f \sim \mathcal{N}(\mu_0, \tau_0 \Sigma_{z_{gi}}^f). \end{aligned} \quad (13)$$

Requiring all clusters in a global component, which may span several video frames, to share a single flow model is too restrictive. Instead we model the flow for each frame independently, requiring only that clusters in frame g assigned to the same component share a common flow model. Our model requires motion of a component to be locally (within a frame) coherent, but allows for large deviations between frames.² This assumption more closely reflects the motion statistics of objects in real videos.

Prior We used data affinities that encourage spatial neighbors not separated by strong intervening contours to

connect to one another. We computed them by independently running the Pb edge detector (Martin et al., 2004) on each video frame and computing $A_{ij} = (1 - b_{ij})^3 \times \mathbf{1}[i, j]$ for each superpixel pair. Here, $0 \leq b_{ij} \leq 1$ is the maximum edge response along a straight line segment connecting the centers of superpixels i, j , and $\mathbf{1}[i, j]$ takes a value of 1 if i and j are spatial neighbors, and 0 otherwise.

Flow-based affinities, as in the earlier toy example, were used to specify the cluster affinity functions. All $\alpha_{1:G}$ and α_0 were set to 10^{-8} . The naive-hddCRP used identical data affinities and hyper-parameters, but used sequential distances between clusters (see Sec. 2.2). The hCRP used sequential affinities to govern both the data and cluster links. For a CRP, the expected number of clusters given N data points is roughly $\alpha \log(N)$. We set $\alpha_{1:G}$ such that the expected number of clusters in a video frame matches the number of observed ground truth clusters, and $\alpha_0 = 1$.

Data link proposals We compare the two data link proposals on 10 frames from the classic “garden” sequence. For each proposal, we ran 3000 iterations of 25 MCMC chains. The results, including MAP samples from the highest and lowest probability chains and log-likelihood trace plots, are summarized in Figure 4. The visualized MAP partitions demonstrate that all chains eventually reach reasonable configurations, but segmentations nevertheless improve qualitatively with increasing model likelihood. This suggests a correspondence between the biases captured by the hddCRP and the statistics of video partitions.

We find that pseudo-Gibbs proposals reach higher probability states more rapidly than prior proposals, and have much lower sensitivity to initialization. Overall, 24 of the 25 pseudo-Gibbs chains reach states that are more probable than the best prior proposal trial. Subsequent experiments thus focus solely on the superior pseudo-Gibbs proposal.

Empirical evaluation We compare our performance against a popular non-probabilistic *hierarchical graph-based video segmentation* (HGVS) algorithm (Grundmann et al., 2010), against the naive-hddCRP variant that was recently used for video co-segmentation (Chiu & Fritz,

¹Chang et al. (2013) also estimate temporal correspondences between superpixels, but we do not utilize this information.

²See the supplement for specific hyper-parameter settings.

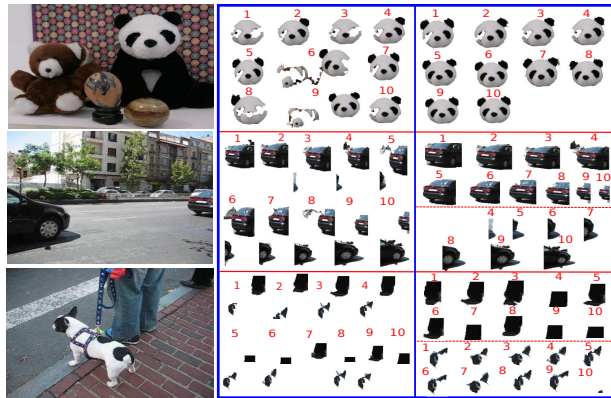


Figure 5: Video segments discovered by the naive-hddCRP and hddCRP. *Left to right*: A representative video frame, segments discovered by the naive-hddCRP and the corresponding segments discovered by hddCRP. Dashed red horizontal lines indicate different segments, and numbers indicate video frame numbers.

2013), and against the hCRP (Teh et al., 2006). For a controlled comparison, all three CRP models use identical likelihoods and hyperparameters. We use the MIT human annotated video dataset (Liu et al., 2008), which contains 9 human annotated videos, to quantitatively measure segmentation performance. We benchmark performance using the first 10 frames of each sequence.

Figure 6 summarizes this experiment. For HGVS the displayed segmentations were produced at 90 percent of the highest hierarchy level, which appears to produce the best visual and quantitative results. For the hddCRP variants, the segmentations correspond to the MAP sample of five MCMC chains, each run for 400 iterations³. We decided to run the samplers for 400 iterations based on the results shown in Figure 4, where a large majority of the pseudo-Gibbs chains converged within the first 300 iterations.

The Rand index was computed by treating the entire video sequence as one spatio-temporal block. This penalizes spatially coherent, but temporally inaccurate, segmentations that exhibit frequent “label switching” between frames. HGVS operates on pixels rather than superpixels and consequently produces finer-scale segmentations. However, these segmentations exhibit large segmentation errors (for instance, the neck and face regions get merged with the background in the second sequence). The hddCRP produces more coherent segmentations and in terms of Rand index, outperforms HGVS on all but one video sequence. The hddCRP also performs substantially better than the hCRP which ignores both superpixel and segment-level correlations; “bag of feature” assumptions are insufficient for this task. The gains over the naive-hddCRP appear to be more modest. However, a closer inspection (Figure 5)

reveals that the hddCRP segments are visually cleaner and more coherent. Additionally, naive-hddCRP often falsely merges visually similar but distinctly moving objects together, while the hddCRP recognizes them as distinct segments. The videos in our dataset have large background regions with no significant motion. Both the hddCRP and the naive-hddCRP models tend to agree on such regions, while disagreeing on smaller foreground objects with distinct motions. Large regions dominate the Rand index, which explains the similar global performance by that metric.

4.2 DISCOURSE SEGMENTATION

Next we consider the problem of discourse segmentation. Given a collection of documents, the goal is to partition each document into a sequence of topically coherent non-overlapping discourse fragments. Previous work by Riedl & Biemann (2012) found that sharing information across documents tends to produce better segmentations, motivating the development of several text segmentation algorithms that exploit document relationships.

We conducted experiments on the *wikielements* dataset (Chen et al., 2009), which consists of 118 Wikipedia articles (at paragraph resolution) describing chemical elements. Although not explicitly made available in the dataset, each article corresponds to a chemical element that is characterized by its chemical properties and has a unique location in the periodic table. Our distance-dependent models are capable of exploiting this additional information to produce better discourse segmentations. As an illustration, consider the alternative problem of clustering articles. Figure 7 illustrates such a clustering where we leverage element properties by defining distances between documents as the Manhattan distance between corresponding element locations in the periodic table. The discovered clustering corresponds well with known element groupings. Discourse segmentation requires clustering the paragraphs describing documents, instead of the documents themselves. Nonetheless, we find that exploiting the periodic table location of each document’s element leads to noticeable performance gains.

We compare two versions of the hddCRP to the naive-hddCRP and hCRP. To encourage topic contiguity, naive-hddCRP and hddCRP allowed paragraphs to either link to themselves or to other paragraphs immediately preceding or succeeding them. We experimented with two affinity functions to capture the intuitions that similar documents tend to present similar topics in similar orders, and that clusters are more likely to be shared among articles about similar elements. The first function (hddCRP1) biased clusters of paragraphs to connect to those that occur at similar locations within other documents. Further, clusters were constrained to connect to only those that were contained in documents with lower atomic numbers. A second variant (hddCRP2) modeled distances between articles using the

³Roughly 6 hours on a 2.3 GHz intel core i7.

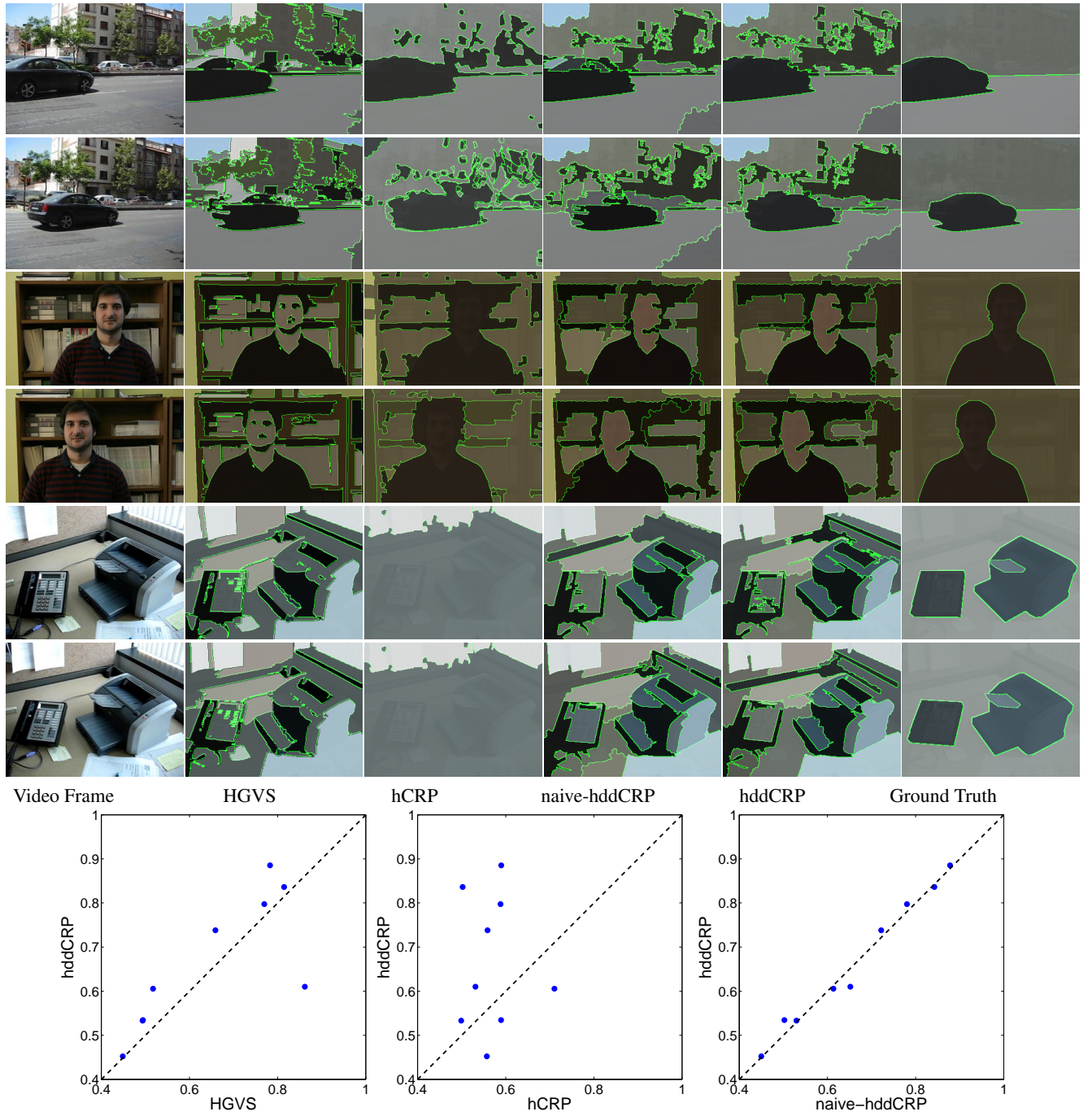


Figure 6: Video segmentation results. The top eight rows show the first and tenth frames of four videos from the MIT human annotated video dataset. *Left to right*: original video frames, segmentations produced by HGVS, hCRP, naive-hddCRP, and hddCRP, and the ground truth segmentations. *Bottom row*: Scatter plots comparing hddCRP, HGVS, naive-hddCRP, and hCRP in terms of Rand index achieved on all nine human annotated videos. Higher scores are better, and more points above the diagonal indicate favorable performance of hddCRP over competitors.

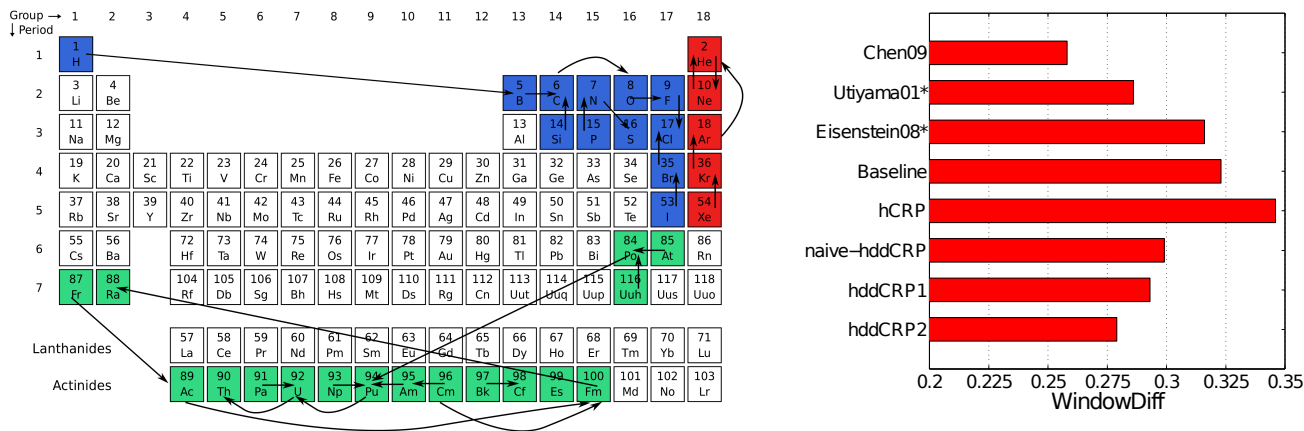


Figure 7: Discourse segmentation results on the *wikelements* dataset. *Left*: A partial visualization of the inferred customer links when clustering Wikipedia articles describing 118 chemical elements. The distance between articles equals the Manhattan distance between their locations in the periodic table. Only three of the nine discovered clusters have been visualized. *Right*: windowDiff scores achieved by competing methods, where lower scores indicate better performance. Asterisks indicate numbers reproduced from Chen et al. (2009).

Manhattan distance between the corresponding element locations in the periodic table, and defined cluster affinities as the logistic decay $f(d) = (1 + \exp(d))^{-1}$ of distances between their corresponding documents. In all cases, we model observed word counts using cluster-specific multinomial distributions with Dirichlet priors. The reported results correspond to the MAP sample of 5 MCMC chains, each run for 400 iterations.

We also compared against established text segmentation methods (Chen et al., 2009; Eisenstein & Barzilay, 2008; Utiyama & Isahara, 2001), and a naïve baseline that groups the entire dataset into one segment. We quantified performance using the windowDiff metric, which slides a window through the text incurring a penalty on discrepancies between the number of segmentation boundaries in the inferred segmentation and a gold standard segmentation. Figure 7 summarizes the performance of the competing models. Both hddCRP1 and hddCRP2 outperform naive-hddCRP and hCRP, showing that our cluster-level affinities capture important additional dataset structure. The hddCRP2 model was superior to all other hddCRP variants, as well as to the specialized text segmentation algorithms of Eisenstein & Barzilay (2008) and Utiyama & Isahara (2001). However, the generalized Mallows model (Chen et al., 2009) achieved the best performance; it is able to both globally bias segment orderings to be similar across related documents, while guaranteeing spatially connected topics. In contrast, the hddCRP weakly constrains segment order through local cluster affinities and while it encourages contiguity, the likelihood may prefer disconnected segments, resulting in a poorer match with the reference segmentation. We nevertheless find it encouraging that the general hddCRP framework, with appropriate affinities, is competitive with specialized text segmentation methods.

5 DISCUSSION AND FUTURE WORK

We have developed a versatile probabilistic model for clustering groups of data with complex structure. Applying it to diverse domains is straightforward: one need only specify appropriate distance functions. Our hierarchical ddCRP defines a valid joint probability distribution for any choice of “distances”, which need not be metrics or have any special properties. Using distances based on pixel locations and optical flow estimates, the hddCRP compares favorably to contemporary video segmentation methods. Using distances based on paragraph order and element positions in the periodic table, it outperforms several established textual discourse segmentation techniques.

While our MCMC inference methods are highly effective for moderate-sized datasets, further innovations will be needed for computational scaling to very large datasets. In cases where training examples of appropriate clusterings are available, we would also like to automatically learn effective hddCRP distance functions.

Acknowledgements

This research supported in part by ONR Award No. N00014-13-1-0644. Portions of this work were completed while S. Ghosh and M. Raptis were at Disney Research, Pittsburgh, PA, USA.

References

- Arandjelović, R. and Zisserman, A. Three things everyone should know to improve object retrieval. In *CVPR*, 2012.
- Blei, D. M. and Frazier, P. I. Distance dependent Chinese restaurant processes. *J. Mach. Learn. Res.*, 12:2461–2488, November 2011.
- Chang, J., Wei, D., and Fisher III, J. W. A Video Representation Using Temporal Superpixels. In *CVPR*, 2013.
- Chen, H., Branavan, S. R. K., Barzilay, R., and Karger, D. R. Content modeling using latent permutations. *J. Artif. Intell. Res. (JAIR)*, 36:129–163, 2009.
- Chiu, W. and Fritz, M. Multi-class video co-segmentation with a generative multi-video model. In *CVPR*, 2013.
- Duan, J. A., Guindani, M., and Gelfand, A. E. Generalized spatial Dirichlet process models. *Biometrika*, 94(4):809–825, 2007.
- Dunson, D. B. and Park, J. Kernel stick-breaking processes. *Biometrika*, 95(2):307–323, 2008.
- Eisenstein, J. and Barzilay, R. Bayesian unsupervised topic segmentation. In *EMNLP*, pp. 334–343, 2008.
- Ghosh, S., Ungureanu, A. B., Sudderth, E. B., and Blei, D. Spatial distance dependent Chinese restaurant processes for image segmentation. In *NIPS*, pp. 1476–1484, 2011.
- Ghosh, S., Sudderth, E. B., Loper, M., and Black, M. J. From deformations to parts: Motion-based segmentation of 3D objects. In *NIPS*, pp. 2006–2014, 2012.
- Green, P. J. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82(4):711–732, 1995.
- Griffin, J. E. and Steel, M. F. J. Order-based dependent Dirichlet processes. *JASA*, 101(473):179–194, March 2006.
- Grundmann, M., Kwatra, V., Han, M., and Essa, I. Efficient hierarchical graph based video segmentation. In *CVPR*, 2010.
- Haines, T. S. F. and Xiang, T. Background subtraction with Dirichlet processes. In *ECCV*, pp. 99–113. Springer, 2012.
- Jain, S. and Neal, R. A split-merge Markov chain Monte Carlo procedure for the Dirichlet process mixture model. *J. of Comput. and Graph. Stats*, 13:158–182, 2004.
- Kim, D. and Oh, A. Accounting for data dependencies within a hierarchical Dirichlet process mixture model. In *CIKM*, pp. 873–878, 2011.
- Lin, D., Grimson, E., and Fisher III, J. W. Construction of dependent Dirichlet processes based on Poisson processes. In *NIPS*, 2010.
- Liu, C., Freeman, W. T., Adelson, E. H., and Weiss, Y. Human-assisted motion annotation. In *CVPR*, 2008.
- MacEachern, S. N. Dependent nonparametric processes. In *Proc. Section on Bayesian Statistical Science*, pp. 50–55. American Statistical Association, 1999.
- Mardia, K. V. and Jupp, P. E. *Directional Statistics*. Wiley Series in Probability and Statistics. Wiley, 2009.
- Martin, D. R., Fowlkes, C.C., and Malik, J. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Trans. PAMI*, 26(5):530–549, 2004.
- Orbanz, P. and Buhmann, J. M. Nonparametric Bayesian image segmentation. *IJCV*, 77:25–45, 2008.
- Pevzner, L. and Hearst, M. A. A critique and improvement of an evaluation metric for text segmentation. *Comput. Linguist.*, 28(1):19–36, March 2002.
- Pitman, J. Combinatorial stochastic processes. Technical Report 621, U.C. Berkeley Department of Statistics, August 2002.
- Rand, W. M. Objective criteria for the evaluation of clustering methods. *JASA*, 66(336):846–850, 1971.
- Rao, V. A. and Teh, Y. W. Spatial normalized gamma processes. In *NIPS*, pp. 1554–1562, 2009.
- Riedl, M. and Biemann, C. How text segmentation algorithms gain from topic models. In *HLT-NAACL*, pp. 553–557, 2012.
- Socher, R., Maas, A., and Manning, C. D. Spectral Chinese restaurant processes: Nonparametric clustering based on similarities. In *AISTATS*, 2011.
- Sudderth, E. B. and Jordan, M. I. Shared segmentation of natural scenes using dependent Pitman-Yor processes. In *NIPS*, pp. 1585–1592, 2008.
- Sudderth, E. B., Torralba, A., Freeman, W. T., and Willsky, A. S. Describing visual scenes using transformed objects and parts. *IJCV*, 77:291–330, 2008.
- Sun, D., Roth, S., and Black, M. J. Secrets of optical flow estimation and their principles. In *CVPR*, pp. 2432–2439. IEEE, June 2010.
- Teh, Y. W., Jordan, M. I., Beal, M. J., and Blei, D. M. Hierarchical Dirichlet processes. *Journal of American Statistical Association*, 25(2):1566–1581, 2006.
- Utiyama, M. and Isahara, H. A statistical model for domain-independent text segmentation. In *ACL*, pp. 499–506, 2001.
- Wang, C. and Blei, D. M. A split-merge MCMC algorithm for the hierarchical Dirichlet process. *arXiv preprint arXiv:1201.1657*, January 2012.

Transformation-based Probabilistic Clustering with Supervision

Siddharth Gopal
Carnegie Mellon University
Pittsburgh, 15213

Yiming Yang
Carnegie Mellon University
Pittsburgh, 15213

Abstract

One of the common problems with clustering is that the generated clusters often do not match user expectations. This paper proposes a novel probabilistic framework that exploits supervised information in a discriminative and transferable manner to generate better clustering of unlabeled data. The supervision is provided by revealing the cluster assignments for *some subset* of the ground truth clusters and is used to learn a transformation of the data such that labeled instances form well-separated clusters with respect to the given clustering objective. This estimated transformation function enables us to fold the remaining unlabeled data into a space where new clusters hopefully match user expectations. While our framework is general, in this paper, we focus on its application to Gaussian and von Mises-Fisher mixture models. Extensive testing on 23 data sets across several application domains revealed substantial improvement in performance over competing methods.

1 INTRODUCTION

Presenting structured and organized views of data to users is crucial for efficient browsing and locating relevant information. Unsupervised clustering techniques have been widely used for discovering latent structures in unlabeled data. Generative clustering models based on Gaussian, von-Mises and Multinomial distributions have emerged as the defacto methods for performing probabilistic clustering.

However, such clustering methods have a fundamental limitation that they do not take into account the user expectations or preferences over different views of data. For example, given a collection of news-stories, one user might prefer to organize them by subject topics such as *Politics*, *Sports*, *Business*, *etc.*, while another user might prefer to see the news-stories grouped by regions such as *U.S.*, *India*, *China*, *etc.* As another example, in a collection of speech

recordings, it is perfectly reasonable to cluster the recordings either by the content of the recording or by the speaker. In both cases, clustering algorithms cannot tell which type of clustering is preferable unless the user's expectation is effectively communicated to the system. This leads to two challenging problems in clustering:

1. How can we effectively inject supervised information to steer the clustering process towards user expectations?
2. If only **some clusters** identified by the user are available as supervision, can the learned user expectations be effectively transferred from the observed clusters to aid in the discovery of **unobserved** (new) clusters in data?

The first problem has been partially addressed by some work in constrained clustering and distance metric learning, although not in sufficient depth (see Section 2). The second problem, to our knowledge, has not been addressed by any work so far. The second problem is particularly important from a practical point of view because realistically users can only label a small set of clusters, whereas the data keeps growing and new clusters are bound to show up. Using a collection of news-stories as a concrete example, if the user identifies three clusters *U.S.*, *India*, and *China* as the supervision, the system should learn that the user is interested in region-based clustering and partition the incoming unlabeled data by regions and discover other new clusters such as *Iraq*, *Japan*, *etc.* In other words, we want the system to generalize the learned user expectation by modeling shared properties between the observed and unobserved clusters.

This paper addresses both challenges by proposing a novel probabilistic framework that assumes the existence of an underlying transformation of data (that is common to the observed and unobserved clusters) which reveals the true user expectations on clustering. Using the supervision in the form of labeled instances and true cluster labels for some subset of groundtruth clusters, we estimate such a transformation. Specifically, we define the transformation function \mathbf{G} in a discriminative manner that maximizes the conditional probability of the cluster label given the transformed instance. Once \mathbf{G} is learned, we apply it to un-

labeled data for clustering in the next step. Because the supervised information is propagated by means of a data transformation we call our framework Transformation-based Clustering with Supervision (TCS).

Our framework is flexible and can be applied to any probabilistic clustering model as long as an appropriate transformation function \mathbf{G} can be defined. In this paper, we explore our framework with two widely used probabilistic clustering models which have different clustering objectives, the Gaussian Mixture model (GM) (Bishop, 2006) and the von Mises-Fisher Mixture model (VM) (Banerjee et al., 2006). For GM, we define \mathbf{G} to be a linear transformation whereas for VM, \mathbf{G} is a linear transformation followed by unit normalization. For both the models we develop efficient optimization algorithms to estimate \mathbf{G} and related parameters.

We conducted thorough evaluations and tested our framework on 23 data sets from different application domains such as time-series, text, handwriting recognition, face recognition, etc. We have observed substantial and consistent improvement in performance (in five clustering metrics) over other competing methods.

2 RELATED WORK

There are two primary areas of related work that use supervision to improve clustering of unlabeled data - Probabilistic Constrained Clustering (PCC) and Distance Metric Learning (DML).

PCC methods (Wagstaff et al., 2001; Basu et al., 2002) try to inject supervision using a probabilistic generative model for the instances. The instances \mathbf{X} consists of both the labeled part \mathbf{X}_L , the unlabeled part \mathbf{X}_U and the cluster assignments $\mathbf{Z} = \{\mathbf{Z}_U \cup \mathbf{Z}_L\}$. The observed cluster assignments $\mathbf{Z}_L \subset \mathbf{Z}$ is used as supervision, i.e., the constraints. The model parameters θ and the latent cluster assignments \mathbf{Z}_U are estimated by maximizing the likelihood under the constraints as:

$$\max_{\theta, \mathbf{Z}_U} \log P(\mathbf{X}|\mathbf{Z}_L, \mathbf{Z}_U, \theta) \quad (1)$$

However, when the task is to detect previously unobserved cluster, this optimization reduces to plain clustering. This can be seen by rewriting the objective as a sum of the labeled objective (which is constant) and the unlabeled objective (which is just standard clustering) as:

$$\max_{\theta_L} \log P(\mathbf{X}_L|\mathbf{Z}_L, \theta_L) + \max_{\theta_U, \mathbf{Z}_U} \log P(\mathbf{X}_U|\mathbf{Z}_U, \theta_U)$$

Clearly, there is no *learning* nor transfer of information from the observed clusters to the unobserved clusters. There are other works in PCC where supervision is represented in the form of pairwise constraints instead of cluster labels, i.e. the constraints are defined as whether two instances should be put in the same cluster or not. These methods optimize eq (1) with an additional penalty term if the constraints are not obeyed. The penalty is introduced in the form of priors (Lu and Leen, 2005), (Basu

et al., 2006) or explicitly modeled using some loss function (Basu et al., 2004), (Bilenko et al., 2004). Despite the different variations in formulating the constraints, PCC methods (Wagstaff et al., 2001; Basu et al., 2002, 2004, 2006; Lu and Leen, 2005) have the same fundamental limitation, i.e., the supervised information from the *observed* clusters is not used to reshape the discovery of *unobserved* clusters. The clustering of the unlabeled part of the data reduces to standard unsupervised clustering.

A natural extension of PCC that addresses some of its limitations is to use a more *Bayesian* approach of sharing parameters across clusters. For example, one could use a Gaussian mixture model where each cluster k has its own mean parameter θ_k , but all clusters share a common covariance matrix Σ . Here the covariance matrix serves as the bridge to transfer information from the observed clusters to the unobserved clusters. One can envision more sophisticated models with common hyperpriors, e.g. a Gaussian hyperprior for the means and an Inverse Wishart hyperprior for the covariances. To our knowledge, such Bayesian revisions of PCC have not been studied in the context of discovering new clusters in unlabeled data (which is the focus of this paper). As we will show in our experiments (where we implemented such a Bayesian PCC model as a baseline), this way of sharing information is not as effective as directly fitting for the cluster labels in the transformed space, which we propose in the TCS framework.

In DML (Blitzer et al., 2005; Xing et al., 2002; Goldberger et al., 2004) the objective is to learn a distance metric that respects the supervision which is provided in the form pairwise constraints. More specifically, given a set of labeled clusters, the distance metric is estimated such that it pulls the within-cluster pairs towards each other and pushes the cross-cluster pairs away from each other. DML methods differ from each other in how the loss functions are defined over the pairwise constraints, such as the hinge loss (Blitzer et al., 2005), Euclidean-distance loss (Xing et al., 2002), log loss (Goldberger et al., 2004), etc. DML has been typically used in the context of nearest-neighbor classification but not in the context of discovering *unobserved* clusters. We argue that existing DML methods have two problems w.r.t to discovering unobserved clusters: Firstly, DML optimizes for pairwise distances and is therefore ‘unaware’ of the clustering criterion (the objective function for clustering) used. This can lead to overfitting, for example, even if the data is optimally clustered, DML would still try to increase inter-cluster distances and decrease intra-cluster distances. Secondly, optimizing for different loss functions (e.g., hinge loss or Euclidean loss) do not necessarily yield a metric that is also optimal for clustering. Explicitly fitting for the cluster-labels that also achieves the optimal clustering objective without resorting to surrogate measures like pairwise constraints is the fundamental difference between our TCS models and other DML methods.

Indirectly related to this paper are a few works in discrim-

Table 1: Likelihood at Local optimum reached by EM vs Likelihood of ground-truth. The Local optimum is better !

Algorithm →	GM	VM
Local optimum	-18115	4.09965e+09
Groundtruth	-18168	4.09906e+09

inative clustering (Krause et al., 2010), (Xu et al., 2004) and constrained spectral clustering (Rangapuram and Hein, 2012), (Wang and Davidson, 2010), (Lu and Carreira-Perpinán, 2008). These former methods use a discriminative objective like that of SVM for clustering, however, cannot handle supervision. The latter methods incorporate supervision in a spectral clustering framework, but however, cannot scale beyond a few thousands of instances, cannot produce probabilistic cluster memberships and do not *directly* optimize for the class-labels. A thorough discussion of the drawbacks of spectral clustering framework is however beyond the scope of this paper (see (Nadler and Galun, 2007) and reference therein). It is also worth mentioning that some other work like (Joulin et al., 2010), (Finley and Joachims, 2005) reformulate the classification problem as a clustering one, but however cannot discover previously unobserved clusters in data.

3 PROPOSED MODEL (TCS)

Any typical clustering task involves making at the least two assumptions - number of clusters (or the prior parameters if using Bayesian non-parametrics) and the distance measure, both of which determine the type of clusters generated. The distance measure in a probabilistic clustering framework is determined by the choice of the distribution for e.g. Euclidean corresponds to Gaussian, cosine-similarity corresponds to von Mises-Fisher etc. Typically, the user’s subjective choice of the probability distribution does not match the ground-truth and the generated clusters do not match user expectations.

To demonstrate this, we ran two popular clustering algorithms - the Gaussian Mixture model (GM) and the von Mises-Fisher mixture model (VM), which use different probability distributions and optimize different likelihoods, on the 20newsgroups dataset ¹ with 20 clusters using the EM algorithm. We compared the likelihoods (the clustering objective) of the algorithms under two settings,

1. The likelihood at a local optimum reached by EM ².
2. The likelihood when the true labels are given, i.e. cluster assignments fixed to the ground-truth.

As table 1 shows, the likelihood obtained at a local optimum is better than groundtruth - the groundtruth is suboptimal ! The clustering algorithm is optimizing for something else other than groundtruth. This means that the user

¹qwone.com/ jason/20NewsGroups/

²The EM algorithm was initialized with the ground-truth cluster assignments

expected clusters can never be recovered. This is a clear case of mismatch between what the user expects and the user specified probability distribution.

To address this issue, we propose to transform the data into another space where the instances are indeed distributed according to user expectations. We use the supervised information to estimate such a transformation i.e. we estimate a transformation function G in a discriminative manner that maximizes the likelihood of observing the given labels (not the data) in the transformed space.

More formally, we are provided supervised training data from K clusters, $\mathcal{S} = \{x_i, t_i\}_{i=1}^N$ where $x_i \in \mathcal{X}$, $t_i \in \{1, 2, \dots, K\}$ and unlabeled examples \mathcal{U} . For convenience define $y_{ik} = I(t_i = k)$. Given a probabilistic generative model using a mixture of K distributions $\{f(x|C_k)\}_{k=1}^K$ where C_k denotes the parameters of cluster K , we estimate G as,

$$\arg \max_G \log P(\mathbf{Y}|\mathbf{C}^{mle}(G), G(\mathbf{X}))$$

$$\text{where } \mathbf{C}^{mle}(G) = \arg \max_{\mathbf{C}} P(G(\mathbf{X})|\mathbf{C}, \mathbf{Y}),$$

$$P(\mathbf{Y}|G(\mathbf{X}), \mathbf{C}^{mle}(G)) = \prod_{i=1}^N \prod_{k=1}^K \left[\frac{f(G(x_i)|C_k^{mle}(G))}{\sum_{k'=1}^K f(G(x_i)|C_{k'}^{mle}(G))} \right]^{y_{ik}}$$

Here \mathbf{C}^{mle} denotes the maximum likelihood estimates of the cluster parameters in the transformed space i.e the cluster parameters that best explain the transformed data. G on the other hand is estimated by maximizing the conditional likelihood of the cluster-labels given \mathbf{C}^{mle} i.e. the transformation that best explains the cluster-labels. Together this ensures that we learn G such that the optimal cluster parameters \mathbf{C}^{mle} also optimally fit the cluster-labels \mathbf{Y} . The transformation G is then applied to \mathcal{U} thereby folding it into a space where hopefully the user specified distribution $f(x|C)$ matches user expectations.

Unlike typical clustering algorithms, our TCS framework has a *learning* component as well i.e. we learn how to discover unobserved clusters from supervision. Since any learning algorithm has chances of overfitting, we add an additional regularization term. Together,

$$\begin{aligned} [\text{OPT1}] \quad & \max_{G, \mathbf{C}} \log P(\mathbf{Y}|\mathbf{C}(G), G(\mathbf{X})) - \gamma \lambda(G) \\ & \text{s.t. } \frac{\partial \log P(G(\mathbf{X})|\mathbf{C}, \mathbf{Y})}{\partial \mathbf{C}} = 0 \end{aligned}$$

where γ is the regularization parameter and λ is the regularization function. Note that the second constraint is another way to say \mathbf{C} is the MLE estimate of $G(\mathbf{X})$. In the following subsections, we discuss how to estimate G with two choices for $f(x|C_k)$ - Gaussian and von Mises-Fisher ³.

³The supplementary material also discusses the extension to Gamma distributions

3.1 TCS WITH GAUSSIAN MIXTURES

We define a simple mixture of K Gaussians with unit variance and cluster means $\{\theta_k\}_{k=1}^K$ over \mathcal{R}^P where

$$f(x|\theta_k) = \frac{1}{\sqrt{2\pi}} \exp(-\|x - \theta_k\|^2)$$

The transformation function is defined as $G(x) = Lx$, a linear transformation using matrix $L \in \mathcal{R}^{P \times P}$. The parameters $\{\theta_k\}_{k=1}^K$ and transformation L is estimated by solving OPT1. Note that the constraint in OPT1 can be written in closed form,

$$\begin{aligned} \text{[OPT1]} \quad & \max_{G, \mathbf{C}} \quad \log P(\mathbf{Y}|\boldsymbol{\theta}, L\mathbf{X}) - \gamma\lambda(G) \\ \text{s.t.} \quad & \theta_k = \frac{1}{n_k} \sum_{i=1}^N y_{ik} Lx_i \quad \forall k \end{aligned}$$

where n_k denotes the number of instances assigned to cluster k . If m_k denotes the mean of instances assigned to cluster k , then OPT1 can be rewritten as,

$$\begin{aligned} \text{[OPT1]} \quad & \max_{G, \mathbf{C}} \quad \log P(\mathbf{Y}|\boldsymbol{\theta}, L\mathbf{X}) - \gamma\lambda(G) \\ \text{s.t.} \quad & \theta_k = Lm_k \quad \forall k \end{aligned}$$

The conditional probability of y_{ik} given the transformed instance Lx_i is,

$$P(y_{ik} = 1|\theta_k, Lx_i) = \frac{e^{-\frac{1}{2}(Lx_i - \theta_k)^\top (Lx_i - \theta_k)}}{\sum_{k'=1}^K e^{-\frac{1}{2}(Lx_i - L\theta_{k'})^\top (Lx_i - L\theta_{k'})}}$$

Letting $\theta_k = Lm_k$ and defining $d_{ik} = x_i - m_k$,

$$P(y_{ik} = 1|\theta_k, Lx_i) = \frac{e^{-\frac{1}{2}d_{ik}^\top L^\top L d_{ik}}}{\sum_{k'=1}^K e^{-d_{ik'}^\top L^\top L d_{ik'}}}$$

By reformulating OPT1⁴ as an optimization problem in terms of $A = L^\top L$, we have a convex semidefinite program,

$$\begin{aligned} \min_A \quad & F(A) = \gamma\lambda(A) + \\ & \sum_{i=1}^N \sum_{k=1}^K \frac{y_{ik}}{2} d_{ik}^\top A d_{ik} + \sum_{i=1}^N \log \left(\sum_{k'=1}^K e^{-\frac{1}{2}d_{ik'}^\top A d_{ik'}} \right) \\ \text{s.t.} \quad & A \succeq 0 \quad \text{where } d_{ik} = (x_i - m_k) \end{aligned}$$

We tried different choices for $\lambda(A)$,

1. $\|A - I\|^2$ (Frobenius Norm from Identity)
2. $\|A\|_2^2$ (Frobenius Norm from zero)
3. $\|A\|_*$ (Nuclear Norm)
4. $\text{trace}(A) - \log(\det(A))$ (Log-det divergence)
5. $\|A - I\|_1$ (Entrywise-L1 Norm)

⁴Due to the lack of space, we defer the detailed derivations to the supplementary material.

Algorithm 1 Accelerated gradient descent for TCS-GM.

- 1: **Input:** $\{\mathbf{X}, \mathbf{Y}\}$, step-length sequence S_t
 - 2: **Initialize:** Define $H_t = I, \beta_t = 1$
 - 3: **while** not converged **do**
 - 4: $A_t = \mathcal{PSD}(H_t - s_t \nabla F(H_t))$
 - 5: $\beta_{t+1} = \frac{1 + \sqrt{1 + 4\beta_t^2}}{2}$
 - 6: $H_{t+1} = A_t + \frac{\beta_t - 1}{\beta_t} (A_t - A_{t-1})$
 - 7: **end while**
 - 8: **Output:** A_t
 - 9: \mathcal{PSD} denotes projection to the positive semidefinite cone
-

Different regularizers favor different kinds of linear transformations, for e.g., Nuclear norm (Ma et al., 2011) favours lower rank solutions, Entrywise-L1 norm favours sparser transformations, the log-det regularizer also prefers lower rank solutions but penalizes sum of log of eigenvalues instead (Davis et al., 2007) etc.

For differentiable regularizers, the optimization can be solved using projected gradient descent where we take a step along the direction of the negative gradient (with the stepsize determined by backtracking line search) and then project the update back into the positive semidefinite cone. We observed that we could significantly improve the speed by using accelerated gradient descent (Beck and Teboulle, 2009) instead (Algorithm 1). For the non-differentiable regularizers we can use projected subgradient instead. These algorithms provably converge to the optimal solution if the step size is appropriately set (Boyd and Vandenberghe, 2004). The gradient of the objective can be succinctly written as,

$$\begin{aligned} \nabla F(A) &= \gamma\lambda'(A) + \sum_{i=1}^N \sum_{k=1}^K (y_{ik} - p_{ik}(A)) d_{ik} d_{ik}^\top \\ &\quad \text{where } p_{ik}(A) = P(y_{ik} | Lx_i, \theta_k) \end{aligned}$$

We found that our customized parallel solver using accelerated projected gradient descent worked much faster than existing SDP solvers. The solution in A recovers L upto any rotation matrix. This is because $A = L^\top L$ can always be rewritten using $A = L^\top (Q^\top Q) L$, for any rotation matrix $Q^{-1} = Q^\top$. However rotating the data corresponds to changing the basis and does not affect the clustering algorithm.

Note that A is very different from the seemingly similar covariance matrix of a Gaussian distribution. Firstly, unlike the covariance matrix, A is not parameter of the distribution and does not make the distribution sum to 1. Secondly, covariance matrices are typically estimated by maximizing $P(\mathbf{X}|\mathbf{Y})$, this includes supervised versions such as linear discriminant analysis (LDA) and Fisher LDA (Hastie et al., 2009). The transformation matrix A on the other hand, is optimized to fit the labels $P(\mathbf{Y}|L\mathbf{X}, \theta)$. Unlike LDA, A does not have a closed form solution.

Algorithm 2 Optimizing $L, \{\mu_k\}_{k=1}^K$ for TCS-VM.

1: **Input:** $\{\mathbf{X}, \mathbf{Y}\}, T$ iterations
2: **Initialize:** L
3: **for** $t = 1, \dots, T$ **do**
4: **update** $\mu_k = \frac{\sum_{i=1}^N y_{ik} n_i}{\sum_{i=1}^N y_{ik}}$ where $n_i = \frac{Lx_i}{\|Lx_i\|}$
5: **update** $L = \arg \min_L \left[\gamma \lambda(L) + \sum_{i=1}^N \sum_k y_{ik} \left[\frac{x_i^\top L^\top \mu_k}{\|Lx_i\|} - \log \left(\sum_{j=1}^K \exp \left(\frac{x_i^\top L^\top \mu_j}{\|Lx_i\|} \right) \right) \right] \right]$
6: **end for**

3.2 TCS WITH VON-MISES FISHER MIXTURES

The von Mises-Fisher (vMF) distribution defines a probability density over points on a unit-sphere. It is parameterized by mean parameter μ and concentration parameter κ , the former defines the direction of the mean and the latter determines the spread of the probability mass around the mean. The density function over $\mathcal{X} \equiv \{x : \|x\| = 1, x \in \mathcal{R}^P\}$, is given by

$$f(x|\mu, \kappa) = \frac{\kappa^{(.5P-1)}}{(2\pi)^{.5P} I_{.5P-1}(\kappa)} \exp(\kappa \mu^\top x)$$

where $I_\nu(a)$ is the modified bessel function of first kind with order ν and argument a .

As in the Gaussian mixtures case, we consider a mixture of K vMF distributions with mean parameters $\{\mu_k\}_{k=1}^K$ and a single concentration parameter κ . Since the support of vMF is only over the unit-sphere, any transformation function should ensure the transformed space still lies on the unit-sphere. Therefore we define the transformation function G as a linear transformation with matrix $L \in \mathcal{R}^{P \times P}$ with normalization,

$$G(x) = \frac{Lx}{\|Lx\|}$$

With this transformation function, the optimization problem OPT1 is a non-convex function in L . Note that the cluster mean parameters $\{\mu_k\}_{k=1}^K$ have a closed form expression for the MLE estimate in terms of L and \mathbf{X} . However unlike the GM case, we do not recommend substituting it into the objective of OPT1 as it leads to computationally intensive expressions. We instead resort to an alternating optimization between μ_k 's and L as shown in Algorithm 2. The optimization step in line 5 is nonconvex in L and can be solved using gradient descent to converge to a locally optimal solution. We found that in practice it worked fine.

4 EXPERIMENTS

4.1 METHODS FOR COMPARISON

We conducted an extensive empirical study of our proposed framework against several competing methods. We tested,

1. **GM** The simple Gaussian Mixture where the data is assumed to be generated from a mixture of K Gaussians with individual means and a single common variance parameter. All parameters are estimated using EM algorithm. Note that this model is unsupervised and cannot use supervision.
2. **TCS-GM** Our proposed TCS using Gaussian mixture model (section 3.1) and $\|A - I\|^2$ regularization, followed by GM on the linearly transformed unlabeled data.
3. **TCS-GM-L2** Our proposed TCS using Gaussian mixture model and plain $\|A\|^2$ regularization, followed by GM on the linearly transformed unlabeled data.
4. **LMNN** (Weinberger and Saul, 2009) One of the most popular local distance metric learning methods that uses hingeloss to push and pull target neighbors. To ensure competitive performance, the number of target neighbors was set to high value - 50. This is followed by GM on the linearly transformed unlabeled data. Note that LMNN is a stronger baseline than other DML methods like Relevant Component Analysis (Shental et al., 2006), Neighborhood Component Analysis (Goldberger et al., 2004), Linear Discriminant Analysis (Fisher, 1936), etc.
5. **PCC** (Bilenko et al., 2004) A popular probabilistic constrained clustering framework. We used a common covariance matrix for all clusters to ensure transfer of information from known to unknown clusters.
6. **Bayesian** We implemented a Bayesian Gaussian mixture model as an additional baseline, where all the cluster means are drawn from a Normal hyperprior and all clusters share a single covariance matrix. We also tried other variants such as cluster-specific covariance matrices with a shared Inverse Wishart hyperprior, but it did not yield any appreciable improvements. We used the familiar constrained EM algorithm (Basu et al., 2002) to derive point estimates for the unknown parameters.
7. **CSP** (Wang and Davidson, 2010) A representative spectral clustering method that can handle supervision in the form of constraints. We use the author recommended method of representing the unlabeled instances with top k (where k is the number of clusters) dimensions generated by the algorithm followed by GM.

We also tested the vMF-based models but due to lack of space we discuss only a part of the results in Section 5.2.1. We defer the complete set of vMF-based results to the supplementary material.

4.2 DATASETS

We tested our framework on 23 datasets (Table 2) from various application domains including timeseries, text, speech, images, etc. A thorough description of the datasets and the details of the feature extraction process is given in the supplementary material.

Table 2: A list of datasets used along with their characteristics.

Domain	Dataset	#Instances	#Dimension	#Classes
Time-series	<i>Australian Signs (Aussign)</i>	2565	352	95
	<i>Character Trajectory (Char)</i>	2858	192	20
	<i>Digital Sports Activity (DSPA)</i>	152	1440	19
	<i>Libras</i>	360	90	15
Text	<i>CNAE</i>	1079	856	9
	<i>K9</i>	2340	21839	20
	<i>TDT4</i>	622	8895	34
	<i>TDT5</i>	6355	20733	125
Handwritten Characters	<i>Penbased Recognition (Penbased)</i>	10992	16	10
	<i>Letter Recognition (Letter)</i>	20000	16	26
	<i>USPS</i>	9298	1984	10
	<i>Binary Alpha Digits (Binalpha)</i>	1404	2480	36
	<i>Optical Recognition (Optrec)</i>	5620	496	10
	<i>MNIST</i>	70000	6076	10
Faces	<i>AT&T faces</i>	400	19964	40
	<i>UMIST</i>	575	10304	20
	<i>Faces96</i>	3016	19375	151
	<i>Labeled Faces in Wild (LFW)</i>	29791	4324	158
Speech	<i>Isolet</i>	7797	617	26
	<i>Wall Street Journal (WSJ)</i>	34942	25	131
Other datasets	<i>Image</i>	2310	18	7
	<i>Vowel</i>	990	10	11
	<i>Leaves</i>	1599	192	100

As in any matrix learning method, for high dimensional datasets, learning (or even storing) a full matrix L is computationally intensive. Previous literature have identified three ways to tackle this issue,

1. Learn a diagonal L instead of full matrix L . This drastically improves the scalability, but at the cost of flexibility in the set of transformations (Weinberger and Saul, 2009).
2. Directly learn a low rank transformation, i.e. $L \in \mathcal{R}^{r \times P}$ where $r \ll P$ instead of a full rank matrix. However, the optimization problem now becomes nonconvex in terms of this low rank L (see (Journée et al., 2010)).
3. Reduce the dimension of the data using Singular Value Decomposition (SVD), followed by learning a full rank matrix on the low dimensional data.

In our experiments, we found that solution (3) worked best. Specifically, dimension reduction using SVD actually **improved** the clustering performance on multiple datasets since it removes the intrinsic noise in the data. This is agreement with several observations in practice (Zha et al., 2001), (Drineas et al., 2004) and in theory (Ding and He, 2004), (Kannan et al., 2004). Therefore, on datasets with more than 200 dimensions, we used SVD to fold the data into a 30 dimensional space. Refer supplementary material (sec 9.1) for detailed experiments on how SVD improves clustering.

4.3 EXPERIMENTAL SETTINGS

For all the experiments, we consider the class-labels associated with the data to be the user expected groundtruth clusters. We randomly partitioned the classes into three sets - training, validation and testing. The methods TCS-GM, TCS-GM-L2 and LMNN use the validation set for tuning the regularization parameter. Note that CSP could not scale on datasets with more than 5000 instances. All the results are reported only on the test-set.

We assume the number of clusters in the unlabeled data is always known, if not, well established techniques like AIC or BIC can be used (finding the right number of clusters is a separate problem that we will not focus on).

We used five clustering metrics for evaluations - Normalized Mutual Information (NMI), Mutual Information (MI), Rand Index (RI), Adjusted Rand Index (ARI) and Purity (refer supplementary material for definitions). All results are averaged over 50 different restarts with Kmeans++ initialization (Arthur and Vassilvitskii, 2007).

5 RESULTS

We present three sets of results that answer the following questions,

1. Does supervision help in clustering better ?
2. By how much do the different methods exploit this supervision ?
3. How does the amount of supervision affect the quality

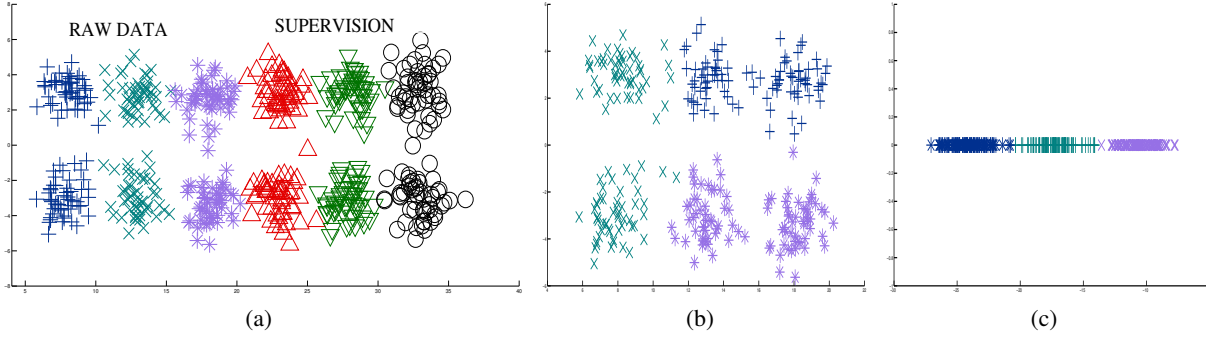


Figure 1: (a) Data from of a mixture of six bimodal Gaussians, (b) Clusters generated by GM on raw data. The generated clusters do not match ground-truth. (c) Clusters generated by TCS-GM. The generated clusters match ground-truth

of clusters generated ?

5.1 ANALYSIS ON SYNTHETIC DATA

First, we show how supervision can be helpful using a small synthetic dataset. We generated data from a mixture of six clusters, where each cluster is a bimodal Gaussian distribution. The clusters lie along the x-axis, where as the modes of the Gaussians are stretched out on the Y-axis (Figure 1a). These are the clusters the user expects to see in the data.

Consider the task of clustering the raw data from first 3 clusters - the darkblue, cyan and purple clusters. Without any supervision, using unsupervised GM results in a clustering show in Figure 1b. Clearly there is confusion between clusters 2 and 3 and the user expectations are not met. On the other hand, if we use TCS-GM with clusters 4, 5, 6 as supervision, we learn a transformation that successfully captures user expectations i.e. the transformation simply collapses all points to the X-axis. When we apply this learnt transformation to the raw data and cluster using GM, we can accurately recover the ground-truth (Fig 1c)

5.2 PERFORMANCE ON REAL-WORLD DATA

Due to lack of space, we report the results only using the NMI metric (the supplementary material contains detailed results using all metrics). Table 3 reports the results of the all supervised and unsupervised models. We defer the results of the vMF-based methods to Section 5.2.1. The results show that our proposed TCS-GM and TCS-GM-L2 achieve the best performance in 19 out of the 23 datasets. We now discuss the results from each domain.

In the time-series domain, *Aussign* and *Libras* are sign-language datasets where each instance represents hand-movement over time, *DSPA* is a human activity recognition dataset where each instance is a human performing one of many predefined actions such as walking, running, etc, *Char* is a handwriting recognition dataset where the instances are (x, y) co-ordinates of pen-tip velocity over time. For all these datasets, the instances are represented as a sequence of vectors (sensor measurements) over time

and the goal is to cluster the instances by the associated classes i.e. handsign (*Aussign*, *Libras*) or activity (*DSPA*) or character (*Char*). To represent each instance as a fixed dimensional vector, we used fast fourier transform to extract the first several high-frequency components of each feature and concatenated them. In this domain TCS-GM-L2 performs the best by showing a 14.2% average improvement over unsupervised GM, followed by TCS-GM with a 9.6% average improvement.

CNAE, *K9*, *TD4*, *TD5* are popular text datasets for classification and clustering (Banerjee et al., 2006). We used the standard bag-of-words with ‘l_{tc}’ term-weighting⁵. On all datasets, only TCS-GM and CSP show any improvement at all. The rest are mostly negatively impacted by supervision. The improvement of TCS-GM is higher than that of CSP.

In the handwritten characters domain (*Penbased*, *Letter*, *USPS*, *Binalpha*, *Optrec* and *MNIST*) each instance is an image-representation of a single character and the task is to cluster the instances characterwise. For feature representation, on the latter four datasets, we used histogram of oriented gradients (HOG) (Srikantan et al., 1996) representation with a patchsize of 2 across 9 different orientations. *Penbased* and *Letter* datasets have predefined set of features such as mean pixel value, edgewise mean, etc. Performance wise, TCS-GM and TCS-GM-L2 achieve the best performance, TCS-GM seems to be more suited to HOG-based features whereas TCS-GM-L2 works better with pixel-based statistical features.

In the face clustering tasks (*AT & T*, *Umist*, *Faces96* and *LFW*) each instance is an image of a single person’s face and the task is to cluster the instances by faces. We represented each instance using HOG features extracted from the image in 9 orientations and varying patchsizes. Both TCS-GM-L2 and LMNN show competitive performance in this task. However, we believe that all datasets except LFW are highly contrived - the images on these other datasets were taken in ideal lighting and posing conditions. The

⁵<http://nlp.stanford.edu/IR-book/html/htmledition/document-and-query-weighting-schemes-1.html>

Table 3: The NMI of the various supervised and unsupervised methods on unnormalized data. The percentage improvement over the unsupervised GM baseline is given in paranthesis. The results of the significance tests between the best method against the other methods on each dataset is denoted by a † for significance at 1% level. **NS** denotes the method could not be scaled to the dataset.

Domain	Dataset	Supervised Learning Method						Unsupervised
		TCS-GM	TCS-GM-L2	LMNN	PCC	BP	CSP	GM
Time series	Aussign	0.89 [†] (8.9%)	0.93 [†] (13.2%)	0.94 (14.7%)	0.78 [†] (-5.1%)	0.84 [†] (2.6%)	0.71 [†] (-13.5%)	0.82 [†]
	Char	0.75 [†] (9.5%)	0.75 (9.8%)	0.67 [†] (-2.3%)	0.69 [†] (0.4%)	0.74 [†] (7.3%)	0.59 [†] (-14.5%)	0.69 [†]
	DSPA	0.69 [†] (1.0%)	0.73 (8.3%)	0.60 [†] (-11.9%)	0.67 [†] (-1.5%)	0.64 [†] (-5.3%)	0.71 [†] (4.6%)	0.68 [†]
	Libras	0.61 [†] (18.7%)	0.64 (25.4%)	0.50 [†] (-2.7%)	0.60 [†] (17.0%)	0.59 [†] (15.6%)	0.45 [†] (-11.7%)	0.51 [†]
	Avg Improvement	9.5%	14.2%	-0.6%	2.7%	5.0%	-8.04%	
Text	CNAE	0.61 (25.4%)	0.24 [†] (-50.7%)	0.31 [†] (-35.3%)	0.43 [†] (-11.1%)	0.48 [†] (-0.8%)	0.59 [†] (21.7%)	0.48 [†]
	K9	0.58 (3.6%)	0.41 [†] (-27.8%)	0.38 [†] (-33.2%)	0.56 [†] (0.2%)	0.51 [†] (-9.1%)	0.58 (3.4%)	0.56 [†]
	TDT4	0.91 (0.8%)	0.69 [†] (-23.3%)	0.90 (0.2%)	0.89 [†] (-0.8%)	0.89 [†] (-0.4%)	0.87 [†] (-2.7%)	0.90 [†]
	TDT5	0.70 (0.6%)	0.69 (0.4%)	0.68 [†] (-2.5%)	0.69 [†] (0.0%)	0.69 (0.3%)	NS	0.69
	Avg Improvement	7.6%	-25.3%	-17.7%	-2.9%	-2.5%	-	
Handwritten Characters	Penbased	0.56 [†] (6.5%)	0.60 (15.7%)	0.57 [†] (9.4%)	0.40 [†] (-23.2%)	0.49 [†] (-6.7%)	NS	0.52 [†]
	Letter	0.48 [†] (36.2%)	0.50 (43.6%)	0.43 [†] (23.6%)	0.27 [†] (-24.5%)	0.37 [†] (6.3%)	NS	0.35 [†]
	USPS	0.84 (3.7%)	0.46 [†] (-44.2%)	0.79 [†] (-3.7%)	0.81 [†] (-1.0%)	0.79 [†] (-3.6%)	NS	0.81 [†]
	Binalpha	0.79 (10.4%)	0.70 [†] (-2.2%)	0.70 [†] (-2.2%)	0.72 [†] (0.8%)	0.68 [†] (-5.4%)	0.67 [†] (-7.2%)	0.72 [†]
	Optrec	0.94 (2.6%)	0.73 [†] (-20.3%)	0.91 [†] (-0.8%)	0.86 [†] (-5.3%)	0.91 [†] (0.2%)	NS	0.91 [†]
	MNIST	0.84 (1.4%)	0.70 [†] (-15.5%)	0.72 [†] (-13.6%)	0.55 [†] (-33.4%)	0.74 [†] (-10.7%)	NS	0.83 [†]
	Avg Improvement	10.1%	-3.8%	2.1%	-14.4%	-3.3%	-	
Faces	AT & T	0.84 [†] (1.1%)	0.88 (5.4%)	0.84 [†] (1.0%)	0.82 [†] (-1.4%)	0.85 [†] (2.2%)	0.78 [†] (-6.1%)	0.83 [†]
	Umist	0.59 [†] (6.1%)	0.74 [†] (33.4%)	0.79 (43.0%)	0.57 [†] (2.7%)	0.56 [†] (1.6%)	0.48 [†] (-13.2%)	0.55 [†]
	Faces96	0.92 [†] (4.1%)	0.93 [†] (4.9%)	0.94 (6.0%)	0.89 [†] (0.1%)	0.89 [†] (0.9%)	0.81 [†] (-8.4%)	0.89 [†]
	LFW	0.39 [†] (36.1%)	0.41 (45.6%)	0.33 [†] (16.1%)	0.31 [†] (9.5%)	0.30 [†] (4.2%)	NS	0.28 [†]
	Avg Improvement	11.9%	22.3%	16.5%	2.7%	2.2%	-	
Speech	Isolet	0.83 (2.2%)	0.80 [†] (-2.5%)	0.81 [†] (-0.5%)	0.75 [†] (-8.5%)	0.83 [†] (1.6%)	NS	0.82 [†]
	WSJ	0.81 (46.5%)	0.81 [†] (46.1%)	0.81 [†] (45.9%)	0.52 [†] (-6.1%)	0.71 [†] (27.4%)	NS	0.56 [†]
	Avg Improvement	24.3%	21.8%	22.7%	-7.3%	14.5%	-	
Other datasets	Image	0.84 (7.2%)	0.40 [†] (-48.8%)	0.49 [†] (-36.5%)	0.60 [†] (-22.9%)	0.70 [†] (-10.4%)	0.75 [†] (-4.5%)	0.78 [†]
	Vowel	0.41 (63.6%)	0.39 [†] (55.7%)	0.35 [†] (36.8%)	0.14 [†] (-46.6%)	0.23 [†] (-7.1%)	0.22 [†] (-13.2%)	0.25 [†]
	Leaves	0.82 [†] (5.9%)	0.82 [†] (5.7%)	0.85 (9.5%)	0.77 [†] (-1.4%)	0.81 [†] (4.8%)	0.73 [†] (-6.4%)	0.78 [†]
	Avg Improvement	25.6%	4.2%	3.2%	-23.7%	-4.2%	-4.2%	

LFW dataset, on the other hand, represents a more *realistic* distribution of images as found on the web. On this dataset, TCS-GM-L2 works best with a 45.6% improvement.

In the speech domain, we tested on two datasets *Isolet*, *WSJ*. In *Isolet* the task is to cluster the instances by the content of the speech (more specifically the letter uttered) and in *WSJ* the task is to cluster by the speaker. The instances are represented using well known audio features such as MFCC's. On both datasets TCS-GM achieves the best performance. The results on *WSJ* particularly highlight the importance of having supervision with a 46% improvement over unsupervised GM.

Image, *Vowel* and *Leaves* are other popularly used classi-

fication datasets. The instances in *Image* and *Leaves* are pictures of outdoor scenes and leaves, whereas in *Vowel* the instances are various utterances of different vowels. On two out of the three datasets, TCS-GM achieves the best results.

To further validate our results, we conducted two-sided significance tests using paired t-test between the best-performing method against the rest of the methods on each dataset. The NMI on the 50 different restarts are considered as observed samples. The null hypothesis is that there is no significance difference between the methods. The results of the testing (Table 3) show that almost all the results are significant.

Table 4: The NMI of TCS-VM with other supervised and unsupervised methods on normalized data. The results of the significance tests between the best method against the other methods on each dataset is denoted by a † for significance at 1% level.

Domain Dataset		Supervised Learning Method							Unsupervised	
		TCS-VM	TCS-GM	TCS-GM-L2	LMNN	PCC	BP	CSP	VM	GM
Text	CNAE	0.905	0.678 [†]	0.678 [†]	0.692 [†]	0.557 [†]	0.462 [†]	0.588 [†]	0.857 [†]	0.669 [†]
	K9	0.638	0.621 [†]	0.443 [†]	0.485 [†]	0.617 [†]	0.584 [†]	0.589 [†]	0.615 [†]	0.616 [†]
	TDT4	0.933	0.916 [†]	0.755 [†]	0.929 [†]	0.914 [†]	0.911 [†]	0.870 [†]	0.930 [†]	0.915 [†]
	TDT5	0.781	0.750 [†]	0.746 [†]	0.707 [†]	0.750 [†]	0.756 [†]	-	0.766 [†]	0.755 [†]

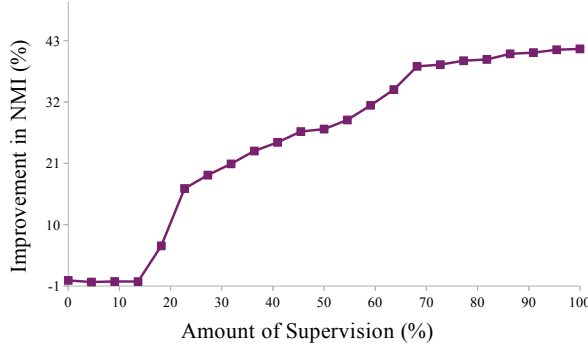


Figure 2: Effect of amount of supervision on the quality of clusters. We plot the improvement in NMI achieved by TSC-GM over baseline GM as we increase the number of training clusters.

5.2.1 PERFORMANCE OF vMF-BASED METHODS

Due to lack of space we present the results of vMF-based models only in the text domain. Normalization to a unit sphere has often shown to work very well for text data, especially vMF-based models have been particularly effective in generating good clusters (Banerjee et al., 2006). We compare the following vMF-based methods on text data,

1. **VM** A mixture of K vMF distributions with individual means and a single common concentration parameter. All parameters are estimated using EM algorithm. Note that this model like GM is unsupervised and cannot use supervision.
2. **TCS-VM** Our proposed TCS using vMF mixture model (section 3.2) and $\|A - I\|^2$ regularization, followed by VM on the transformed unlabeled data.

For data representation, we used the bag-of-words with ‘lrc’ term-weighting and svd-based dimension reduction to 30 dimensions, followed by a normalization to unit-sphere. The results are shown in Table 4. For an informative comparison we also included the results of the other methods - TCS-GM, TCS-GM-L2, LMNN, PCC, BP, CSP and GM. On all text datasets the proposed TCS-VM model performs best. In the complete set of results for all domains for normalized data (presented in the supplementary material), our

proposed TCS models achieve the best performance in 20 out of the 23 datasets.

5.3 EFFECT OF AMOUNT OF SUPERVISION

We analyze how the quality of clusters generated depend on the amount of supervision provided. We used a subset of the WSJ dataset with 25 training and 25 testing clusters for this task.

Figure 2 plots the improvement in NMI achieved by TCS-GM over baseline GM as we increase the number of training clusters. Initially there is no improvement in performance, but as training clusters increase, there is a gradual improvement in performance, until it reaches a saturation level. This shows that (a) there is some minimum amount of supervision needed to see any improvement (b) providing more supervision does not increase performance indefinitely but saturates at certain level. This kind of curve is typical of most machine learning algorithms.

6 CONCLUSION

In this paper we proposed a novel framework that can exploit supervision to generate *better* clustering of unlabeled data. By learning a common underlying transformation function, our framework is able to successfully generalize the observed clusters to discover new clusters that match user expectations. We explored two instantiations of our framework with Gaussian and von Mises-Fisher mixture models. For both the models we developed fast optimization algorithms to estimate the model parameters. Our extensive testing on 23 datasets provide strong empirical support in favour of our proposed framework. In future, we would like to adapt our framework to spectral clustering based methods.

7 Acknowledgements

We thank the anonymous reviewers for their helpful suggestions and feedback. This work is supported in part by the National Science Foundation (NSF) under grant IIS 1216282.

References

David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth*

- annual ACM-SIAM symposium on Discrete algorithms, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.
- Arindam Banerjee, Inderjit S Dhillon, Joydeep Ghosh, and Suvrit Sra. Clustering on the unit hypersphere using von mises-fisher distributions. *Journal of Machine Learning Research*, 6(2): 1345, 2006.
- S. Basu, A. Banerjee, and R. Mooney. Semi-supervised clustering by seeding. In *ICML*, pages 19–26, 2002.
- S. Basu, M. Bilenko, and R.J. Mooney. A probabilistic framework for semi-supervised clustering. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 59–68. ACM, 2004.
- S. Basu, M. Bilenko, A. Banerjee, and R.J. Mooney. Probabilistic semi-supervised clustering with constraints. *Semi-supervised learning*, pages 71–98, 2006.
- Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.
- M. Bilenko, S. Basu, and R.J. Mooney. Integrating constraints and metric learning in semi-supervised clustering. In *Proceedings of the twenty-first international conference on Machine learning*, page 11. ACM, 2004.
- C.M. Bishop. *Pattern recognition and machine learning*. 2006.
- John Blitzer, Kilian Q Weinberger, and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. In *NIPS*, pages 1473–1480, 2005.
- Stephen Poythress Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.
- Jason V Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S Dhillon. Information-theoretic metric learning. In *Proceedings of the 24th international conference on Machine learning*, pages 209–216. ACM, 2007.
- Chris Ding and Xiaofeng He. K-means clustering via principal component analysis. In *Proceedings of the twenty-first international conference on Machine learning*, page 29. ACM, 2004.
- Petros Drineas, Alan Frieze, Ravi Kannan, Santosh Vempala, and V Vinay. Clustering large graphs via the singular value decomposition. *Machine learning*, 56(1-3):9–33, 2004.
- T. Finley and T. Joachims. Supervised clustering with support vector machines. In *Proceedings of the 22nd international conference on Machine learning*, pages 217–224. ACM, 2005.
- Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936.
- Jacob Goldberger, Sam Roweis, Geoff Hinton, and Ruslan Salakhutdinov. Neighbourhood components analysis. 2004.
- Trevor Hastie, Robert Tibshirani, Jerome Friedman, T Hastie, J Friedman, and R Tibshirani. *The elements of statistical learning*, volume 2. Springer, 2009.
- Armand Joulin, Francis Bach, and Jean Ponce. Discriminative clustering for image co-segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1943–1950. IEEE, 2010.
- Michel Journée, Francis Bach, P-A Absil, and Rodolphe Sepulchre. Low-rank optimization on the cone of positive semidefinite matrices. *SIAM Journal on Optimization*, 20(5):2327–2351, 2010.
- Ravi Kannan, Santosh Vempala, and Adrian Vetta. On clusterings: Good, bad and spectral. *Journal of the ACM (JACM)*, 51(3): 497–515, 2004.
- Andreas Krause, Pietro Perona, and Ryan G Gomes. Discriminative clustering by regularized information maximization. In *NIPS*, pages 775–783, 2010.
- Z. Lu and T. Leen. Semi-supervised learning with penalized probabilistic clustering. *NIPS*, 17:849–856, 2005.
- Zhengdong Lu and Miguel A Carreira-Perpinán. Constrained spectral clustering through affinity propagation. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.
- Shiqian Ma, Donald Goldfarb, and Lifeng Chen. Fixed point and bregman iterative methods for matrix rank minimization. *Mathematical Programming*, 128(1-2):321–353, 2011.
- Boaz Nadler and Meirav Galun. Fundamental limitations of spectral clustering. *NIPS*, 19:1017, 2007.
- Syama S Rangapuram and Matthias Hein. Constrained 1-spectral clustering. In *International Conference on Artificial Intelligence and Statistics*, pages 1143–1151, 2012.
- Noam Shental, Tomer Hertz, Daphna Weinshall, and Misha Pavel. Adjustment learning and relevant component analysis. In *Computer Vision ECCV 2002*, pages 776–790. Springer, 2006.
- Geetha Srikantan, Stephen W Lam, and Sargur N Srihari. Gradient-based contour encoding for character recognition. *Pattern Recognition*, 29(7):1147–1160, 1996.
- Kiri Wagstaff, Claire Cardie, Seth Rogers, and Stefan Schrödl. Constrained k-means clustering with background knowledge. In *ICML*, volume 1, pages 577–584, 2001.
- Xiang Wang and Ian Davidson. Flexible constrained spectral clustering. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 563–572. ACM, 2010.
- Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *The Journal of Machine Learning Research*, 10:207–244, 2009.
- Eric P Xing, Andrew Y Ng, Michael I Jordan, and Stuart Russell. Distance metric learning, with application to clustering with side-information. *NIPS*, 15:505–512, 2002.
- Linli Xu, James Neufeld, Bryce Larson, and Dale Schuurmans. Maximum margin clustering. In *NIPS*, pages 1537–1544, 2004.
- Hongyuan Zha, Xiaofeng He, Chris Ding, Ming Gu, and Horst D Simon. Spectral relaxation for k-means clustering. In *NIPS*, volume 1, pages 1057–1064, 2001.

Understanding the Complexity of Lifted Inference and Asymmetric Weighted Model Counting

Eric Gribkoff

University of Washington
eagribko@cs.uw.edu

Guy Van den Broeck

KU Leuven, UCLA
guyvdb@cs.ucla.edu

Dan Suciu

University of Washington
suciu@cs.uw.edu

Abstract

In this paper we study lifted inference for the Weighted First-Order Model Counting problem (WFOMC), which counts the assignments that satisfy a given sentence in first-order logic (FOL); it has applications in Statistical Relational Learning (SRL) and Probabilistic Databases (PDB). We present several results. First, we describe a lifted inference algorithm that generalizes prior approaches in SRL and PDB. Second, we provide a novel dichotomy result for a non-trivial fragment of FO CNF sentences, showing that for each sentence the WFOMC problem is either in PTIME or #P-hard in the size of the input domain; we prove that, in the first case our algorithm solves the WFOMC problem in PTIME, and in the second case it fails. Third, we present several properties of the algorithm. Finally, we discuss limitations of lifted inference for symmetric probabilistic databases (where the weights of ground literals depend only on the relation name, and not on the constants of the domain), and prove the impossibility of a dichotomy result for the complexity of probabilistic inference for the entire language FOL.

1 INTRODUCTION

Weighted model counting (WMC) is a problem at the core of many reasoning tasks. It is based on the model counting or #SAT task (Gomes et al., 2009), where the goal is to count assignments that satisfy a given logical sentence. WMC generalizes model counting by assigning a weight to each assignment, and computing the *sum of their weights*. WMC has many applications in AI and its importance is increasing. Most notably, it underlies state-of-the-art probabilistic inference algorithms for Bayesian networks (Darwiche, 2002; Sang et al., 2005; Chavira and Darwiche,

2008), relational Bayesian networks (Chavira et al., 2006) and probabilistic programs (Fierens et al., 2011).

This paper is concerned with weighted *first-order* model counting (WFOMC), where we sum the weights of assignments that satisfy a sentence in finite-domain first-order logic. Again, this reasoning task underlies efficient algorithms for probabilistic reasoning, this time for popular representations in statistical relational learning (SRL) (Getoor and Taskar, 2007), such as Markov logic networks (Van den Broeck et al., 2011; Gogate and Domingos, 2011) and probabilistic logic programs (Van den Broeck et al., 2014). Moreover, WFOMC uncovers a deep connection between AI and database research, where query evaluation in *probabilistic databases* (PDBs) (Suciu et al., 2011) essentially considers the same task. A PDB defines a probability, or weight, for every possible world, and each database query is a sentence encoding a set of worlds, whose combined probability we want to compute.

Early on, the disconnect between compact relational representations of uncertainty, and the intractability of inference at the ground, propositional level was noted, and efforts were made to exploit the relational structure for inference, using so-called *lifted inference* algorithms (Poole, 2003; Kersting, 2012). SRL and PDB algorithms for WFOMC all fall into this category. Despite these commonalities, there are also important differences. SRL has so far considered *symmetric* WFOMC problems, where relations of the same type are assumed to contribute equally to the probability of a world. This assumption holds for certain queries on SRL models, such as single marginals and partition functions, but fails for more complex conditional probability queries. These break lifted algorithms based on symmetric WFOMC (Van den Broeck and Darwiche, 2013). PDBs, on the other hand, have considered the *asymmetric* WFOMC setting from the start. While there are many semantics for PDBs, the most common models are tuple-independent PDBs, which assign each tuple a distinct probability, many tuples have probability zero, and no symmetries can be expected. However, current asymmetric WFOMC algorithms (Dalvi and Suciu, 2012) suffer from a major limitation of

their own, in that they can only count models of sentences in monotone disjunctive normal form (MDNF) (i.e., DNF without negation). Such sentences represent unions of conjunctive database queries (UCQ). WFOMC encodings of SRL models almost always fall outside this class.

The present work seeks to upgrade a well-known PDB algorithm for asymmetric WFOMC (Dalvi and Suciu, 2012) to the SRL setting, by enabling it to count models of arbitrary sentences in conjunctive normal form (CNF). This permits its use for lifted SRL inference with arbitrary soft or hard evidence, or equivalently, probabilistic database queries with negation. Our first contribution is this algorithm, which we call **Lift^R**, and is presented in Section 3.

Although **Lift^R** has clear practical merits, we are in fact motivated by fundamental theoretical questions. In the PDB setting, our algorithm is known to come with a sharp complexity guarantee, called the *dichotomy theorem* (Dalvi and Suciu, 2012). By only looking at the structure of the first-order sentence (i.e., the database query), the algorithm reports failure when the problem is #P-hard (in terms of data complexity), and otherwise guarantees to solve it in time polynomial in the domain (i.e., database) size. It can thus precisely classify MDNF sentences as being tractable or intractable for asymmetric WFOMC. Whereas several complexity results for symmetric WFOMC exist (Van den Broeck, 2011; Jaeger and Van den Broeck, 2012), the complexity of asymmetric WFOMC for SRL queries with evidence is still poorly understood. Our second and main contribution, presented in Section 4, is a *novel dichotomy result* over a small but non-trivial fragment of CNFs. We completely classify this class of problems as either computable in polynomial time or #P-hard. This represents a first step towards proving the following conjecture: **Lift^R** provides a dichotomy for asymmetric WFOMC on arbitrary CNF sentences, and therefore perfectly classifies all related SRL models as tractable or intractable for conditional queries.

As our third contribution, presented in Section 5, we illustrate the algorithm with examples that show its application to common probabilistic models. We discuss the capabilities of **Lift^R** that are not present in other lifted inference techniques.

As our fourth and final contribution, in Section 6, we discuss extensions of our algorithm to symmetric WFOMC, but also show the impossibility of a dichotomy result for arbitrary first-order logic sentences.

2 BACKGROUND

We begin by introducing the necessary background on relational logic and weighted model counting.

2.1 RELATIONAL LOGIC

Throughout this paper, we will work with the relational fragment of first-order logic (FOL), which we now briefly review. An *atom* $P(t_1, \dots, t_n)$ consists of predicate P/n of arity n followed by n arguments, which are either *constants* or *logical variables* $\{x, y, \dots\}$. A *literal* is an atom or its negation. A *formula* combines atoms with logical connectives and quantifiers \exists and \forall . A *substitution* $[a/x]$ replaces all occurrences of x by a . Its application to formula F is denoted $F[a/x]$. A formula is a sentence if each logical variable x is enclosed by a $\forall x$ or $\exists x$. A formula is *ground* if it contains no logical variables. A *clause* is a universally quantified disjunction of literals. A *term* is an existentially quantified conjunction of literals. A *CNF* is a conjunction of clauses, and a *DNF* is a disjunction of terms. A *monotone* CNF or DNF contains no negation symbols. As usual, we drop the universal quantifiers from the CNF syntax.

The semantics of sentences are defined in the usual way (Hinrichs and Genesereth, 2006). An interpretation, or world, I that satisfies sentence Δ is denoted by $I \models \Delta$, and represented as a set of literals. Our algorithm checks properties of sentences that are undecidable in general FOL, but decidable, with the following complexity, in the CNF fragment we investigate.

Theorem 2.1. (Sagiv and Yannakakis, 1980) (Farré et al., 2006) *Checking whether logical implication $Q \Rightarrow Q'$ or equivalence $Q \equiv Q'$ holds between two CNF sentences is Π_2^P -complete.*

2.2 WEIGHTED MODEL COUNTING

Weighted model counting was introduced as a propositional reasoning problem.

Definition 2.2 (WMC). *Given a propositional sentence Δ over literals \mathcal{L} , and a weight function $w : \mathcal{L} \rightarrow \mathbb{R}^{\geq 0}$, the weighted model count (WMC) is*

$$\text{WMC}(\Delta, w) = \sum_{I \models \Delta} \prod_{\ell \in I} w(\ell).$$

We will consider its generalization to *weighted first-order model counting* (WFOMC), where Δ is now a sentence in relational logic, and \mathcal{L} consists of all ground first-order literals for a given domain of constants.

The WFOMC task captures query answering in probabilistic database. Take for example the database

Prof(Ane) : 0.9 Prof(Charlie) : 0.1
Student(Bob) : 0.5 Student(Charlie) : 0.8
Advises(Ane, Bob) : 0.7 Advises(Bob, Charlie) : 0.1

and the UCQ (monotone DNF) query

$$Q = \exists x, \exists y, \text{Prof}(x) \wedge \text{Advises}(x, y) \wedge \text{Student}(y).$$

If we set $\Delta = Q$ and w to map each literal to its probability in the database, then our query answer is

$$\Pr(Q) = \text{WFOMC}(\Delta, w) = 0.9 \cdot 0.7 \cdot 0.5 = 0.315.$$

We refer to the general case above as *asymmetric* WFOMC, because it allows $w(\text{Prof}(\text{Anne}))$ to be different from $w(\text{Prof}(\text{Charlie}))$. We use *symmetric* WFOMC to refer to the special case where w simplifies into two weight functions w^*, \bar{w}^* that map *predicates* to weights, instead of literals, that is

$$w(\ell) = \begin{cases} w^*(P) & \text{when } \ell \text{ is of the form } P(c) \\ \bar{w}^*(P) & \text{when } \ell \text{ is of the form } \neg P(c) \end{cases}$$

Symmetric WFOMC no longer directly captures PDBs. Yet it can still encode many SRL models, including parfactor graphs (Poole, 2003), Markov logic networks (MLNs) (Richardson and Domingos, 2006) and probabilistic logic programs (De Raedt et al., 2008). We refer to (Van den Broeck et al., 2014) for the details, and show here the following example MLN.

$$2 \quad \text{Prof}(x) \wedge \text{Advises}(x, y) \Rightarrow \text{Student}(y)$$

It states that the probability of a world increases by a factor e^2 with every pair of people x, y for which the formula holds. Its WFOMC encoding has Δ equal to

$$\forall x, \forall y, F(x, y) \Leftrightarrow [\text{Prof}(x) \wedge \text{Advises}(x, y) \Rightarrow \text{Student}(y)]$$

and weight functions w^*, \bar{w}^* such that $w^*(F) = e^2$ and all other predicates map to 1.

Answering an SRL query Q given evidence E , that is, $\Pr(Q|E)$, using a symmetric WFOMC encoding, generally requires solving two WFOMC tasks:

$$\Pr(Q|E) = \frac{\text{WFOMC}(Q \wedge E \wedge \Delta, w)}{\text{WFOMC}(E \wedge \Delta, w)}$$

Symmetric WFOMC problems are strictly more tractable than asymmetric ones. We postpone the discussion of this observation to Section 5, but already note that all theories Δ with up to two logical variables per formula support *domain-lifted* inference (Van den Broeck, 2011), which means that any WFOMC query runs in time polynomial in the domain size (i.e., number of constants). For conditional probability queries, even though fixed-parameter complexity bounds exist that use symmetric WFOMC (Van den Broeck and Darwiche, 2013), the actual underlying reasoning task is asymmetric WFOMC, whose complexity we investigate for the first time.

Finally, we make three simplifying observations. First, SRL query Q and evidence E typically assign values to random variables. This means that the query and evidence can be absorbed into the asymmetric weight function, by setting the weight of literals disagreeing with Q or E to

zero. We hence compute:

$$\Pr(Q|E) = \frac{\text{WFOMC}(\Delta, w_{Q \wedge E})}{\text{WFOMC}(\Delta, w_E)}$$

This means that our complexity analysis for a given encoding Δ applies to both numerator and denominator for arbitrary Q and E , and that polytime WFOMC for Δ implies polytime $\Pr(Q|E)$ computation. The converse is not true, since it is possible that both WFOMC calls are #P-hard, but their ratio is in PTIME. Second, we will from now on assume that Δ is in CNF. The WFOMC encoding of many SRL formalisms is already in CNF, or can be reduced to it (Van den Broeck et al., 2014). For PDB queries that are in monotone DNF, we can simply compute $\Pr(Q) = 1 - \Pr(\neg Q)$, which reduces to WFOMC on a CNF. Moreover, by adjusting the probabilities in the PDB, this CNF can also be made monotone. Third, we will assume that $w(\ell) = 1 - w(\neg \ell)$, which can always be achieved by normalizing the weights.

Under these assumptions, we can simply refer to $\text{WFOMC}(Q, w)$ as $\Pr(Q)$, to Q as the CNF query, to $w(\ell)$ as the probability $\Pr(\ell)$, and to the entire weight function w as the PDB. This is in agreement with notation in the PDB literature.

3 ALGORITHM **Lift^R**

We present here the lifted algorithm **Lift^R** (pronounced *lift-ER*), which, given a CNF formula Q computes $\Pr(Q)$ in polynomial time in the size of the PDB, or fails. In the next section we provide some evidence for its completeness: under certain assumptions, if **Lift^R** fails on formula Q , then computing $\Pr(Q)$ is #P-hard in the PDB size.

3.1 DEFINITIONS

An *implicate* of Q is some clause C s.t. the logical implication $Q \Rightarrow C$ holds. C is a *prime implicate* if there is no other implicate C' s.t. $C' \Rightarrow C$.

A *connected component* of a clause C is a minimal subset of its atoms that have no logical variables in common with the rest of the clause. If some prime implicate C has more than one connected component, then we can write it as:

$$C = D_1 \vee D_2 \vee \dots \vee D_m$$

where each D_i is a clause with distinct variables. Applying distributivity, we write Q in *union-CNF* form:

$$Q = Q_1 \vee Q_2 \vee \dots \vee Q_m$$

where each Q_i is a CNF with distinct variables.

We check for disconnected prime implicates $D_1 \vee D_2$ where both D_1 and D_2 subsume some clause of Q . Intuitively, this means that when we apply inclusion/exclusion to the union-CNF, the resulting queries are simpler. The search

for D_1, D_2 can proceed using some standard inference algorithm, e.g. resolution. By Theorem 2.1, this problem is Π_2^P -complete in the size of the query Q , but independent of the PDB size.

A set of *separator variables* for a query $Q = \bigwedge_{i=1}^k C_i$ is a set of variables $x_i, i = 1, k$ such that, (a) for each clause C_i, x_i occurs in all atoms of C_i , and (b) any two atoms (not necessarily in the same clause) referring to the same relation R have their separator variable on the same position.

3.2 PREPROCESSING

We start by transforming Q (and PDB) such that:

1. No constants occur in Q .
2. If all the variables in Q are x_1, x_2, \dots, x_k , then every relational atom in Q (positive or negated) is of the form $R(x_{i_1}, x_{i_2}, \dots)$ such that $i_1 < i_2 < \dots$.

Condition (1) can be enforced by *shattering* Q w.r.t. its variables. Condition (2) can be enforced by modifying both the query Q and the database, in a process called *ranking* and described in the appendix. Here, we illustrate ranking on an example. Consider the query:

$$Q = (R(x, y) \vee S(x, y)) \wedge (\neg R(x, y) \vee \neg S(y, x))$$

Define $R_1(x, y) \equiv R(x, y) \wedge (x < y)$; $R_2(x) \equiv R(x, x)$; $R_3(y, x) \equiv R(x, y) \wedge (x > y)$. Define similarly S_1, S_2, S_3 . Given a PDB with relations R, S , we define a new PDB' over the six relations by setting $\Pr(R_1(a, b)) = \Pr(R(a, b))$ when $a < b$, $\Pr(R_1(a, b)) = 0$ when $a > b$, $\Pr(R_2(a)) = \Pr(R(a, a))$, etc. Then, the query Q over PDB is equivalent to the following query over PDB':

$$\begin{aligned} & (R_1(x, y) \vee S_1(x, y)) \wedge (\neg R_1(x, y) \vee \neg S_3(x, y)) \wedge \\ & (R_2(x) \vee S_2(x)) \wedge (\neg R_2(x) \vee \neg S_2(x)) \wedge \\ & (R_3(x, y) \vee S_3(x, y)) \wedge (\neg R_3(x, y) \vee \neg S_1(x, y)) \end{aligned}$$

3.3 ALGORITHM DESCRIPTION

Algorithm **Lift^R**, given in Figure 1, proceeds recursively on the structure of the CNF query Q . When it reaches ground atoms, it simply looks up their probabilities in the PDB. Otherwise, it performs the following sequence of steps.

First, it tries to express Q as a union-CNF. If it succeeds, and if the union can be partitioned into two sets that do not share any relational symbols, $Q = Q_1 \vee Q_2$, then it applies a *Decomposable Disjunction*:

$$\Pr(Q) = 1 - (1 - \Pr(Q_1))(1 - \Pr(Q_2))$$

Otherwise, it applies the *Inclusion/Exclusion* formula:

$$\Pr(Q) = - \sum_{s \subseteq [m]} (-1)^{|s|} \Pr(\bigwedge_{i \in s} Q_i)$$

However, before computing the recursive probabilities, our algorithm first checks for equivalent expressions, i.e. it

Algorithm **Lift^R**

Input: Ranked and shattered query Q
 Probabilistic DB with domain D

Output: $\Pr(Q)$

```

1 Step 0: If  $Q$  is a single ground literal  $\ell$ , return
           its probability  $\Pr(\ell)$  in PDB
2 Step 1: Write  $Q$  as a union-CNF:  $Q = Q_1 \vee Q_2 \vee \dots \vee Q_m$ 
3 Step 2: If  $m > 1$  and  $Q$  can be partitioned into two
           sets  $Q = Q' \vee Q''$  with disjoint relation symbols,
           return  $1 - (1 - \Pr(Q_1)) \cdot (1 - \Pr(Q_2))$ 
4           /* Decomposable Disjunction */
5 Step 3: If  $Q$  cannot be partitioned, return
            $\sum_{s \subseteq [m]} (-1)^{|s|} \Pr(\bigwedge_{i \in s} Q_i)$ 
6           /* Inclusion/Exclusion - perform cancellations
           before recursion */
7 Step 4: Write  $Q$  in CNF:  $Q = C_1 \wedge C_2 \wedge \dots \wedge C_k$ 
8 Step 5: If  $k > 1$ , and  $Q$  can be partitioned into two
           sets  $Q = Q' \wedge Q''$  with disjoint relation symbols,
           return  $\Pr(Q_1) \cdot \Pr(Q_2)$ 
9           /* Decomposable Conjunction */
10 Step 6: If  $Q$  has a separator variable, return
            $\prod_{a \in D} \Pr(C_1[a/x_1] \wedge \dots \wedge C_k[a/x_k])$ 
11           /* Decomposable Universal Quantifier */
12 Otherwise FAIL
```

Figure 1: Algorithm for Computing $\Pr(Q)$

checks for terms s_1, s_2 in the inclusion/exclusion formula such that $\bigwedge_{i \in s_1} Q_i \equiv \bigwedge_{i \in s_2} Q_i$: in that case, these terms either cancel out, or add up (and need be computed only once). We show in Section 5.4 the critical role that the cancellation step plays for the completeness of the algorithm. To check cancellations, the algorithm needs to check for equivalent CNF expressions. This can be done using some standard inference algorithm (recall from Theorem 2.1 that this problem is Π_2^P -complete in the size of the CNF expression).

If neither of the above steps apply, then the algorithm checks if Q can be partitioned into two sets of clauses that do not share any common relation symbols. In that case, $Q = Q' \wedge Q''$, and its probability is computed using a *Decomposable Conjunction*:

$$\Pr(Q) = \Pr(Q') \cdot \Pr(Q'')$$

Finally, if none of the above cases apply to the CNF query $Q = C_1 \wedge C_2 \wedge \dots \wedge C_k$, then the algorithm tries to find a set of separator variables x_1, \dots, x_k (one for each clause). If it finds them, then the probability is given by a *Decomposable Universal Quantifier*:

$$\Pr(Q) = \prod_{a \in \text{Domain}} \Pr(C_1[a/x_1] \wedge \dots \wedge C_k[a/x_k])$$

We prove our first main result:

Theorem 3.1. *One of the following holds: (1) either **Lift^R** fails on Q , or (2) for any domain size n and a PDB consisting of probabilities for the ground tuples, **Lift^R** computes*

$\Pr(Q)$ in polynomial time in n .

Proof. (Sketch) The only step of the algorithm that depends on the domain size n is the decomposable universal quantifier step; this also reduces by 1 the arity of every relation symbol, since it substitutes it by the same constant a . Therefore, the algorithm runs in time $O(n^k)$, where k is the largest arity of any relation symbol. We note that the constant behind $O(\dots)$ may be exponential in the size of the query Q . \square

4 MAIN COMPLEXITY RESULT

In this section we describe our main technical result of the paper: that the algorithm is complete when restricted to a certain class of CNF queries.

We first review a prior result, to put ours in perspective. (Dalvi and Suciu, 2012) define an algorithm for Monotone DNF (called Unions Of Conjunctive Queries), which can be adapted to Monotone CNF; that adaptation is equivalent to **Lift^R** restricted to Monotone CNF queries. (Dalvi and Suciu, 2012) prove:

Theorem 4.1. *If algorithm **Lift^R** FAILS on a Monotone CNF query Q , then computing $\Pr(Q)$ is #P-hard.*

However, the inclusion of negations in our query language increases significantly the difficulty of analyzing query complexities. Our major technical result of the paper extends Theorem 4.1 to a class of CNF queries with negation.

Define a *Type-1* query to be a CNF formula where each clause has at most two variables denoted x, y , and each atom is of one of the following three kinds:

- Unary symbols $R_1(x), R_2(x), R_3(x), \dots$
- Binary symbols $S_1(x, y), S_2(x, y), \dots$
- Unary symbols $T_1(y), T_2(y), \dots$

or the negation of these symbols.

Our main result is:

Theorem 4.2. *For every Type-1 query Q , if algorithm **Lift^R** FAILS then computing $\Pr(Q)$ is #P-hard.*

The proof is a significant extension of the techniques used by (Dalvi and Suciu, 2012) to prove Theorem 4.1; we give a proof sketch in Section 7 and include the full proof in the appendix.

5 PROPERTIES OF **Lift^R**

We now describe several properties of **Lift^R**, and the relationship to other lifted inference formalisms.

5.1 NEGATIONS CAN LOWER THE COMPLEXITY

The presence of negations can lower a query's complexity, and our algorithm exploits this. To see this, consider the following query

$$Q = (\text{Tweets}(x) \vee \neg \text{Follows}(x, y)) \\ \wedge (\text{Follows}(x, y) \vee \neg \text{Leader}(y))$$

The query says that if x follows anyone then x tweets, and that everybody follows the leader¹.

Our goal is to compute the probability $\Pr(Q)$, knowing the probabilities of all atoms in the domain. We note that the two clauses are dependent (since both refer to the relation `Follow`), hence we cannot simply multiply their probabilities; in fact, we will see that if we remove all negations, then the resulting query is #P-hard; the algorithm described by (Dalvi and Suciu, 2012) would immediately get stuck on this query. Instead, **Lift^R** takes advantage of the negation, by first computing the prime implicate:

$$\text{Tweets}(x) \vee \neg \text{Leader}(y)$$

which is a disconnected clause (the two literals use disjoint logical variables, x and y respectively). After applying distributivity we obtain:

$$Q \equiv (Q \wedge (\text{Tweets}(x))) \vee (Q \wedge (\neg \text{Leader}(y))) \\ \equiv Q_1 \vee Q_2$$

and **Lift^R** applies the inclusion-exclusion formula:

$$\Pr(Q) = \Pr(Q_1) + \Pr(Q_2) - \Pr(Q_1 \wedge Q_2)$$

After simplifying the three queries, they become:

$$Q_1 = (\text{Follows}(x, y) \vee \neg \text{Leader}(y)) \\ \wedge (\text{Tweets}(x))$$

$$Q_2 = (\text{Tweets}(x) \vee \neg \text{Follows}(x, y)) \\ \wedge (\neg \text{Leader}(y))$$

$$Q_1 \wedge Q_2 = (\text{Tweets}(x)) \wedge (\neg \text{Leader}(y))$$

The probability of Q_1 can now be obtained by multiplying the probabilities of its two clauses; same for the other two queries. As a consequence, our algorithm computes the probability $\Pr(Q)$ in polynomial time in the size of the domain and the PDB.

If we remove all negations from Q and rename the predicates we get the following query:

$$h_1 = (R(x) \vee S(x, y)) \wedge (S(x, y) \vee T(y))$$

(Dalvi and Suciu, 2012) proved that computing the probability of h_1 is #P-hard in the size of the PDB. Thus, the query Q with negation is *easy*, while h_1 is hard, and our algorithm takes advantage of this by applying resolution.

¹To see this, rewrite the query as $(\text{Follows}(x, y) \Rightarrow \text{Tweets}(x)) \wedge (\text{Leader}(y) \Rightarrow \text{Follows}(x, y))$.

5.2 ASYMMETRIC WEIGHTS CAN INCREASE THE COMPLEXITY

(Van den Broeck, 2011) has proven that any query with at most two logical variables per clause is domain-liftable. Recall that this means that one can compute its probability in PTIME in the size of the domain, in the symmetric case, when all tuples in a relation have the same probability. However, queries with at most two logical variables per clause can become #P-hard when computed over asymmetric probabilities, as witnessed by the query h_1 above.

5.3 COMPARISON WITH PRIOR LIFTED FO-CIRCUITS

(Van den Broeck et al., 2011; Van den Broeck, 2013) introduce FO d-DNNF circuits, to compute symmetric WFOMC problems. An FO d-DNNF is a circuit whose nodes are one of the following: decomposable conjunction ($Q_1 \wedge Q_2$ where Q_1, Q_2 do not share any common predicate symbols), deterministic-disjunction ($Q_1 \vee Q_2$ where $Q_1 \wedge Q_2 \equiv \text{false}$), inclusion-exclusion, decomposable universal quantifier (a type of $\forall x, Q(x)$), and deterministic automorphic existential quantifier. The latter is an operation that is specific only to structures with symmetric weights, and therefore does not apply to our setting. We prove that our algorithm can compute all formulas that admit an FO d-DNNF circuit.

Fact 5.1. *If Q admits an FO d-DNNF without a deterministic automorphic existential quantifier, then Lift^R computes $\Pr(Q)$ in PTIME in the size of the PDB.*

The proof is immediate by noting that all other node types in the FO d-DNNF have a corresponding step in Lift^R , except for deterministic disjunction, which our algorithm computes using inclusion-exclusion: $\Pr(Q_1 \vee Q_2) = \Pr(Q_1) + \Pr(Q_2) - \Pr(Q_1 \wedge Q_2) = \Pr(Q_1) + \Pr(Q_2)$ because $Q_1 \wedge Q_2 \equiv \text{false}$.

5.4 CANCELLATIONS IN INCLUSION/EXCLUSION

We now look at a more complex query. First, let us denote four simple queries:

$$\begin{aligned} q_0 &= (R(x_0) \vee S_1(x_0, y_0)) \\ q_1 &= (S_1(x_1, y_1) \vee S_2(x_1, y_1)) \\ q_2 &= (S_2(x_2, y_2) \vee S_3(x_2, y_2)) \\ q_3 &= (S_3(x_3, y_3) \vee T(y_3)) \end{aligned}$$

(Dalvi and Suciu, 2012) proved that their conjunction, i.e. the query $h_3 = q_0 \wedge q_1 \wedge q_2 \wedge q_3$, is #P-hard in data complexity. Instead of h_3 , consider:

$$Q_W = (q_0 \vee q_1) \wedge (q_0 \vee q_3) \wedge (q_2 \vee q_3)$$

There are three clauses sharing relation symbols, hence we cannot apply a decomposable conjunction. However, each

clause is disconnected, for example q_0 and q_1 do not share logical variables, and we can thus write Q_W as a disjunction. After removing redundant terms:

$$Q_W = (q_0 \wedge q_2) \vee (q_0 \wedge q_3) \vee (q_1 \wedge q_3)$$

Our algorithm applies the inclusion/exclusion formula:

$$\begin{aligned} \Pr(Q_W) &= \Pr(q_0 \wedge q_2) + \Pr(q_0 \wedge q_3) + \Pr(q_1 \wedge q_3) \\ &\quad - \Pr(q_0 \wedge q_2 \wedge q_3) - \Pr(q_0 \wedge q_1 \wedge q_3) - \Pr(q_0 \wedge \dots \wedge q_3) \\ &\quad + \Pr(q_0 \wedge \dots \wedge q_3) \end{aligned}$$

At this point our algorithm performs an important step: it cancels out the last two terms of the inclusion/exclusion formula. Without this key step, no algorithm could compute the query in PTIME, because the last two terms are precisely h_3 , which is #P-hard. To perform the cancellation the algorithm needs to first check which FOL formulas are equivalent, which, as we have seen, is decidable for our language (Theorem 2.1). Once the equivalent formulas are detected, the resulting expressions can be organized in a lattice, as shown in Figure 2, and the coefficient of each term in the inclusion-exclusion formula is precisely the lattice's Möbius function (Stanley, 1997).

6 EXTENSIONS AND LIMITATIONS

We describe here an extension of Lift^R to symmetric WFOMC, and also prove that a complete characterization of the complexity of all FOL queries is impossible.

6.1 SYMMETRIC WFOMC

Many applications of SRL require weighted model counting for FOL formulas over PDBs where the probabilities are associated to relations rather than individual tuples. That is, $\text{Friend}(a, b)$ has the same probability, independently of the constants a, b in the domain. In that symmetric WFOMC case, the model has a large number of symmetries (since the probabilities are invariant under permutations of constants), and lifted inference algorithms may further exploit these symmetries. (Van den Broeck, 2013) employ one operator that is specific to symmetric probabilities, called *atom counting*, which is applied to a unary predicate $R(x)$ and iterates over all possible worlds of that predicate. Although there are 2^n possible worlds for R , by conditioning on any world, the probability will depend only on the cardinality k of R , because of the symmetries. Therefore, the system iterates over $k = 0, n$, and adds the conditional probabilities multiplied by $\binom{n}{k}$. For example, consider the following query:

$$H = (\neg R(x) \vee S(x, y) \vee \neg T(y)) \quad (1)$$

Computing the probabilities of this query is #P-hard (Theorem 4.2). However, if all tuples $R(a)$ have the same probability $r \in [0, 1]$, and similarly tuples in S, T have proba-

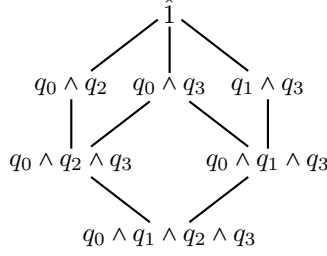


Figure 2: Lattice for Q_w . The bottom query is #P-hard, yet all terms in the inclusion/exclusion formula that contain this term cancel out, and $\Pr(Q_w)$ is computable in PTIME.

bilities s, t , then one can check that²

$$\Pr(H) = \sum_{k,l=0,n} r^k \cdot (1-r)^{n-k} \cdot t^l \cdot (1-t)^{n-l} \cdot (1-s^{kl})$$

Denote **Sym-Lift^R** the extension of **Lift^R** with a deterministic automorphic existential quantifier operator. The question is whether this algorithm is complete for computing the probabilities of queries over PDBs with symmetric probabilities. Folklore belief was that this existential quantifier operator was the only operator required to exploit the extra symmetries available in PDBs with symmetric probabilities. For example, all queries in (Van den Broeck et al., 2011) that can be computed in PTIME over symmetric PDBs have the property that, if one removes all unary predicates from the query, then the residual query can be computed in PTIME over asymmetric PDBs.

We answer this question in the negative. Consider the following query:

$$Q = (S(x_1, y_1) \vee \neg S(x_1, y_2) \vee \neg S(x_2, y_1) \vee S(x_2, y_2))$$

Here, we interpret $S(x, y)$ as a *typed relation*, where the values x and y are from two disjoint domains, of sizes n_1, n_2 respectively, in other words, $S \subseteq [n_1] \times [n_2]$.

Theorem 6.1. *We have that*

- $\Pr(Q)$ can be computed in time polynomial in the size of a symmetric PDB with probability p as $\Pr(Q) = f(n_1, n_2) + g(n_1, n_2)$ where:

$$f(0, n_2) = 1 \quad f(n_1, n_2) = \sum_{k=1}^{n_1} p^{kn_2} g(n_1 - k, n_2)$$

$$g(n_1, 0) = 1 \quad g(n_1, n_2) = \sum_{\ell=1}^{n_2} (1-p)^{n_1\ell} f(n_1, n_2 - \ell)$$

- **Sym-Lift^R** fails to compute Q .

The theorem shows that new operators will be required for symmetric WFOMC. We note that it is currently open whether computing $\Pr(Q)$ is #P-hard in the case of asymmetric WFOMC.

²Conditioned on $|R| = k$ and $|T| = l$, the query is true if S contains at least one pair $(a, b) \in R \times T$.

Proof. Denote D_x, D_y the domains of the variables x and y . Fix a relation $S \subseteq D_1 \times D_2$. We will denote $a_1, a_2, \dots \in D_1$ elements from the domain of the variable x , and $b_1, b_2, \dots \in D_2$ elements from the domain of the variable y . For any a, b , define $a < b$ if $(a, b) \in S$, and $a > b$ if $(a, b) \notin S$; in the latter case we also write $b < a$. Then, (1) for any a, b , either $a < b$ or $b < a$, (2) $<$ is a partial order on the disjoint union of the domains D_1 and D_2 iff S satisfies the query Q . The first property is immediate. To prove the second property, notice that Q states that there is no cycle of length 4: $x_1 < y_2 < x_2 < y_1 < x_1$. By repeatedly applying resolution between Q with itself, we derive that there are no cycles of length 6, 8, 10, etc. Therefore, $<$ is transitive, hence a partial order. Any finite, partially ordered set has a minimal element, i.e. there exists z s.t. $\forall x, x \not< z$. Let Z be the set of all minimal elements, and denote $X = D_1 \cap Z$ and $Y = D_2 \cap Z$. Then exactly one of X or Y is non-empty, because if both were non-empty then, for $a \in X$ and $b \in Y$ we have either $a < b$ or $a > b$ contradicting their minimality. Assuming $X \neq \emptyset$, we have (a) for all $a \in X$ and $b \in D_2$, $(a, b) \in S$, and (b) Q is true on the relation $S' = (D_1 - X) \times D_2$. This justifies the recurrence formula for $\Pr(Q)$. \square

6.2 THE COMPLEXITY OF ARBITRARY FOL QUERIES

We conjecture that, over asymmetric probabilities (asymmetric WFOMC), our algorithm is complete, in the sense that whenever it fails on a query, then the query is provably #P-hard. Notice that **Lift^R** applies only to a fragment of FOL, namely to CNF formulas without function symbols, and where all variables are universally quantified. We present here an impossibility result showing that a complete algorithm cannot exist for general FOL queries. We use for that a classic result by Trakhtenbrot (Libkin, 2004):

Theorem 6.2 (Finite satisfiability). *The problem: “given a FOL sentence Φ , check whether there exists a finite model for Φ ” is undecidable.*

From here we obtain:

Theorem 6.3. *There exists no algorithm that, given any FOL sentence Q checks whether $\Pr(Q)$ can be computed*

in PTIME in the asymmetric PDB size.

Proof. By reduction from the finite satisfiability problem. Fix the hard query H in Eq.(1), for which the counting problem is #P-hard. Recall that H uses the symbols R, S, T . Let Φ be any formula over a disjoint relational vocabulary (i.e. it doesn't use R, S, T). We will construct a formula Q , such that computing $\Pr(Q)$ is in PTIME iff Φ is unsatisfiable in the finite: this proves the theorem. To construct Q , first we modify Φ as follows. Let $P(x)$ be another fresh, unary relational symbol. Rewrite Φ into Φ' as follows: replacing every $(\exists x.\Gamma)$ with $(\exists x.P(x) \wedge \Gamma)$ and every $(\forall x.\Gamma)$ with $(\forall x.P(x) \Rightarrow \Gamma)$ (this is not equivalent to the guarded fragment of FOL); leave the rest of the formula unchanged. Intuitively, Φ' checks if Φ is true on the substructure defined by the domain elements that satisfy P . More precisely: for any database instance I , Φ' is true on I iff Φ is true on the substructure of I defined by the domain elements that satisfy $P(x)$. Define the query $Q = (H \wedge \Phi')$. We now prove the claim.

If Φ is unsatisfiable then so is Φ' , and therefore $\Pr(Q) = 0$ is trivially computable in PTIME.

If Φ is satisfiable, then fix any deterministic database instance I that satisfies Φ ; notice that I is deterministic, and $I \models \Phi$. Let J be any probabilistic instance over the vocabulary for H over a domain disjoint from I . Define $P(x)$ as follows: $P(a)$ is true for all domain elements $a \in I$, and $P(b)$ is false for all domain elements $b \in J$. Consider now the probabilistic database $I \cup J$. (Thus, $P(x)$ is also deterministic, and selects the substructure I from $I \cup J$; therefore, Φ' is true in $I \cup J$.) We have $\Pr(Q) = \Pr(H \wedge \Phi') = \Pr(H)$, because Φ' is true on $I \cup J$. Therefore, computing $\Pr(Q)$ is #P-hard. Notice the role of P : while I satisfies Φ , it is not necessarily the case that $I \cup J$ satisfies Φ . However, by our construction we have ensured that $I \cup J$ satisfies Φ' . \square

7 PROOF OF THEOREM 4.2

The proof of Theorem 4.2 is based on a reduction from the #PP2-CNF problem, which is defined as follows. Given two disjoint sets of Boolean variables X_1, \dots, X_n and Y_1, \dots, Y_n and a bipartite graph $E \subseteq [n] \times [n]$, count the number of satisfying truth assignments $\#\Phi$ to the formula: $\Phi = \bigwedge_{(i,j) \in E} (X_i \vee Y_j)$. (Provan and Ball, 1983) have shown that this problem is #P-hard.

More precisely, we prove the following: given any Type-1 query Q on which the algorithm **Lift^R** fails, we can reduce the #PP2-CNF problem to computing $\Pr(Q)$ on a PDB with domain size n . The reduction consists of a combinatorial part (the construction of certain gadgets), and an algebraic part, which makes novel use of the concepts of algebraic independence (Yu, 1995) and annihilating polynomials (Kayal, 2009). We include the latter in the appendix,

and only illustrate here the former on a particular query of Type-1.

We illustrate the combinatorial part of the proof on the following query Q :

$$(R(x) \vee \neg S(x, y) \vee T(y)) \wedge (\neg R(x) \vee S(x, y) \vee \neg T(y))$$

To reduce Φ to the problem of computing $\Pr(Q)$, we construct a structure with unary predicates R and T and binary predicate S , with active domain $[n]$.

We define the tuple probabilities as follows. Letting $x, y, a, b \in (0, 1)$ be four numbers that will be specified later, we define:

$$\begin{aligned} \Pr(R(i)) &= x \\ \Pr(T(j)) &= y \\ \Pr(S(i, j)) &= \begin{cases} a & \text{if } (i, j) \in E \\ b & \text{if } (i, j) \notin E \end{cases} \end{aligned}$$

Note this PDB does not have symmetric probabilities: in fact, over structures with symmetric probabilities one can compute $\Pr(Q)$ in PTIME.

Let θ denote a valuation of the variables in Φ . Let E_θ denote the event $\forall i. (R(i) = \text{true} \iff \theta(X_i) = \text{true}) \wedge \forall j. (T(j) = \text{true} \iff \theta(Y_j) = \text{true})$.

E_θ completely fixes the unary predicates R and T and leaves S unspecified. Given E_θ , each Boolean variable corresponding to some $S(x, y)$ is now independent of every other $S(x', y')$. In general, given an assignment of $R(i)$ and $T(j)$, we examine the four formulas that define the probability that the query is true on (i, j) : $F_1 = Q[R(i) = 0, T(j) = 0]$, $F_2 = Q[R(i) = 0, T(j) = 1]$, $F_3 = Q[R(i) = 1, T(j) = 0]$, $F_4 = Q[R(i) = 1, T(j) = 1]$.

For Q , F_1, F_2, F_3, F_4 are as follows:

$$F_1 = \neg S(i, j) \quad F_2 = F_3 = \text{true} \quad F_4 = S(i, j)$$

Denote f_1, f_2, f_3, f_4 the arithmetization of these Boolean formulas:

$$\begin{aligned} f_1 &= \begin{cases} 1 - a & \text{if } (i, j) \in E \\ 1 - b & \text{if } (i, j) \notin E \end{cases} \\ f_4 &= \begin{cases} a & \text{if } (i, j) \in E \\ b & \text{if } (i, j) \notin E \end{cases} \end{aligned}$$

Note that $f_2 = f_3 = 1$ and do not change $\Pr(Q)$.

Define the parameters k, l, p, q of E_θ as k = number of i 's s.t. $R(i) = \text{true}$, l = number of j 's s.t. $T(j) = \text{true}$, p = number of $(i, j) \in E$ s.t. $R(i) = T(j) = \text{true}$, q = number of $(i, j) \in E$ s.t. $R(i) = T(j) = \text{false}$.

Let $N(k, l, p, q)$ = the number of θ 's that have parameters k, l, p, q . If we knew all $(n+1)^2(m+1)^2$ values of $N(k, l, p, q)$, we could recover $\#\Phi$ by summing over $N(k, l, p, q)$ where $q = 0$. That is, $\#\Phi = \sum_{k, l, p} N(k, l, p, 0)$.

We now describe how to solve for $N(k, l, p, q)$, completing the hardness proof for $\Pr(Q)$.

We have $\Pr(E_\theta) = x^k(1-x)^{n-k}y^l(1-y)^{n-l}$ and $\Pr(Q|E_\theta) = a^p(1-a)^q b^{kl-p}(1-b)^{(n-k)(n-l)-q}$. Combined, these give the following expression for $\Pr(Q)$:

$$\begin{aligned} \Pr(Q) &= \sum_{\theta} \Pr(Q|E_\theta) \Pr(E_\theta) \\ &= (1-b)^{n^2} (1-x)^n (1-y)^n \sum_{k,l,p,q} T \end{aligned} \quad (1)$$

where:

$$\begin{aligned} T &= N(k, l, p, q) * (a/b)^p [(1-a)/(1-b)]^q \\ &\quad [x/(1-b)^n (1-x)]^k [y/(1-b)^n \\ &\quad (1-y)]^l [b(1-b)]^{kl} \\ &= N(k, l, p, q) * A^p B^q X^k Y^l C^{kl} \end{aligned} \quad (2)$$

Equations (1) and (2) express $\Pr(Q)$ as a polynomial in X, Y, A, B, C with unknown coefficients $N(k, l, p, q)$. Our reduction is the following: we choose $(n+1)^2(m+1)^2$ values for the four parameters $x, y, a, b \in (0, 1)$, consult an oracle for $\Pr(Q)$ for these settings of the parameters, then solve a linear system of $(n+1)^2(m+1)^2$ equations in the unknowns $N(k, l, p, q)$. The crux of the proof consists of showing that the matrix of the system is non-singular: this is far from trivial, in fact had we started from a PTIME query Q then the system *would* be singular. Our proof consists of two steps (1) prove that we can choose X, Y, A, B independently, in other words that the mapping $(x, y, a, b) \mapsto (X, Y, A, B)$ is locally invertible (has a non-zero Jacobian), and (2) prove that there exists a choice of $(n+1)^2(m+1)^2$ values for (X, Y, A, B) such that the matrix of the system is non-singular: then, by (1) it follows that we can find $(n+1)^2(m+1)^2$ values for (x, y, a, b) that make the matrix non-singular, completing the proof. For our particular example, Part (1) can be verified by direct computations (see Section A.3); for general queries this requires Section A.12. Part (2) for this query is almost as general as for any query and we show it in Section A.2.

8 RELATED WORK

The algorithm and complexity results of (Dalvi and Suciu, 2012), which apply to positive queries, served as the starting point for our investigation of asymmetric WFOMC with negation. See (Suciu et al., 2011) for more background on their work. The tuple-independence assumption of PDBs presents a natural method for modeling asymmetric WFOMC. Existing approaches for PDBs can express complicated correlations (Jha et al., 2010; Jha and Suciu, 2012) but only consider queries without negation.

Close in spirit to the goals of our work are (Van den Broeck, 2011) and (Jaeger and Van den Broeck, 2012). They introduce a formal definition of lifted inference and describe a

powerful knowledge compilation technique for WFOMC. Their completeness results for first-order knowledge compilation on a variety of query classes motivate our exploration of the complexity of lifted inference. (Cozman and Polastro, 2009) analyze the complexity of probabilistic description logics.

Other investigations of evidence in lifted inference include (Van den Broeck and Davis, 2012), who allow arbitrary hard evidence on unary relations, (Bui et al., 2012), who allow asymmetric soft evidence on a single unary relation, and (Van den Broeck and Darwiche, 2013), who allow evidence of bounded Boolean rank. Our model allows entirely asymmetric probabilities and evidence.

9 CONCLUSION

Our first contribution is the algorithm **Lift^R** for counting models of arbitrary CNF sentences over asymmetric probabilistic structures. Second, we prove a novel dichotomy result that completely classifies a subclass of CNFs as either PTIME or #P-hard. Third, we describe capabilities of **Lift^R** not present in prior lifted inference techniques. Our final contribution is an extension of our algorithm to symmetric WFOMC and a discussion of the impossibility of establishing a dichotomy for all first-order logic sentences.

Acknowledgements

This work was partially supported by ONR grant #N00014-12-1-0423, NSF grants IIS-1115188 and IIS-1118122, and the Research Foundation-Flanders (FWO-Vlaanderen).

References

- Hung B Bui, Tuyen N Huynh, and Rodrigo de Salvo Braz. Exact lifted inference with distinct soft evidence on every object. In *AAAI*, 2012.
- Mark Chavira and Adnan Darwiche. On probabilistic inference by weighted model counting. *Artificial Intelligence*, 172(6-7):772–799, April 2008.
- Mark Chavira, Adnan Darwiche, and Manfred Jaeger. Compiling relational Bayesian networks for exact inference. *International Journal of Approximate Reasoning*, 42(1-2):4–20, May 2006.
- Fabio Gagliardi Cozman and Rodrigo Bellizia Polastro. Complexity analysis and variational inference for interpretation-based probabilistic description logics. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 117–125. AUAI Press, 2009.
- Nilesh Dalvi and Dan Suciu. The dichotomy of probabilistic inference for unions of conjunctive queries. *Journal of the ACM (JACM)*, 59(6):30, 2012.

- Adnan Darwiche. A logical approach to factoring belief networks. *Proceedings of KR*, pages 409–420, 2002.
- Luc De Raedt, Paolo Frasconi, Kristian Kersting, and Stephen Muggleton, editors. *Probabilistic inductive logic programming: theory and applications*. Springer-Verlag, Berlin, Heidelberg, 2008. ISBN 3-540-78651-1, 978-3-540-78651-1.
- Carles Farré, Werner Nutt, Ernest Teniente, and Toni Urpí. Containment of conjunctive queries over databases with null values. In *Database Theory–ICDT 2007*, pages 389–403. Springer, 2006.
- Daan Fierens, Guy Van den Broeck, Ingo Thon, Bernd Gutmann, and Luc De Raedt. Inference in probabilistic logic programs using weighted CNF’s. In *Proceedings of UAI*, pages 211–220, July 2011.
- L. Getoor and B. Taskar, editors. *An Introduction to Statistical Relational Learning*. MIT Press, 2007.
- Vibhav Gogate and Pedro Domingos. Probabilistic theorem proving. In *Proceedings of UAI*, pages 256–265, 2011.
- Carla P Gomes, Ashish Sabharwal, and Bart Selman. Model counting. *Handbook of Satisfiability*, 185:633–654, 2009.
- Timothy Hinrichs and Michael Genesereth. Herbrand logic. Technical Report LG-2006-02, Stanford University, Stanford, CA, 2006.
- Manfred Jaeger and Guy Van den Broeck. Liftability of probabilistic inference: Upper and lower bounds. In *Proceedings of the 2nd International Workshop on Statistical Relational AI*, 2012.
- Abhay Jha and Dan Suciu. Probabilistic databases with markovviews. *Proceedings of the VLDB Endowment*, 5(11):1160–1171, 2012.
- Abhay Jha, Vibhav Gogate, Alexandra Meliou, and Dan Suciu. Lifted inference seen from the other side: The tractable features. In *Advances in Neural Information Processing Systems 23*, pages 973–981. 2010.
- Neeraj Kayal. The complexity of the annihilating polynomial. In *Computational Complexity, 2009. CCC’09. 24th Annual IEEE Conference on*, pages 184–193. IEEE, 2009.
- Kristian Kersting. Lifted probabilistic inference. In *Proceedings of European Conference on Artificial Intelligence (ECAI)*, 2012.
- Leonid Libkin. *Elements of Finite Model Theory*. Springer, 2004. ISBN 3-540-21202-7.
- David Poole. First-order probabilistic inference. In *IJCAI*, volume 3, pages 985–991. Citeseer, 2003.
- J Scott Provan and Michael O Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM Journal on Computing*, 12(4):777–788, 1983.
- Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine learning*, 62(1-2):107–136, 2006.
- Yehoshua Sagiv and Mihalis Yannakakis. Equivalences among relational expressions with the union and difference operators. *Journal of the ACM (JACM)*, 27(4):633–655, 1980.
- T. Sang, P. Beame, and H. Kautz. Solving Bayesian networks by weighted model counting. In *Proceedings of AAAI*, volume 1, pages 475–482, 2005.
- Richard P. Stanley. *Enumerative Combinatorics*. Cambridge University Press, 1997.
- Dan Suciu, Dan Olteanu, Christopher Ré, and Christoph Koch. Probabilistic databases. *Synthesis Lectures on Data Management*, 3(2):1–180, 2011.
- Guy Van den Broeck. On the completeness of first-order knowledge compilation for lifted probabilistic inference. In *NIPS*, pages 1386–1394, 2011.
- Guy Van den Broeck. *Lifted Inference and Learning in Statistical Relational Models*. PhD thesis, Ph. D. Dissertation, KU Leuven, 2013.
- Guy Van den Broeck and Adnan Darwiche. On the complexity and approximation of binary evidence in lifted inference. In *Advances in Neural Information Processing Systems*, pages 2868–2876, 2013.
- Guy Van den Broeck and Jesse Davis. Conditioning in first-order knowledge compilation and lifted probabilistic inference. In *Proceedings of AAAI*, 2012.
- Guy Van den Broeck, Nima Taghipour, Wannes Meert, Jesse Davis, and Luc De Raedt. Lifted probabilistic inference by first-order knowledge compilation. In *IJCAI*, pages 2178–2185, 2011.
- Guy Van den Broeck, Wannes Meert, and Adnan Darwiche. Skolemization for weighted first-order model counting. In *Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning (KR)*, 2014.
- Jie-Tai Yu. On relations between jacobians and minimal polynomials. *Linear algebra and its applications*, 221:19–29, 1995.

Interactive Learning from Unlabeled Instructions

Jonathan Grizou

Inria Bordeaux Sud-Ouest, France
jonathan.grizou@inria.fr

Iñaki Iturrate

CNBI, EPFL, Switzerland
inaki.iturrate@epfl.ch

Luis Montesano

I3A, Univ. of Zaragoza, Spain
montesano@unizar.es

Pierre-Yves Oudeyer

Inria Bordeaux Sud-Ouest, France
pierre-yves.oudeyer@inria.fr

Manuel Lopes

Inria Bordeaux Sud-Ouest, France
manuel.lopes@inria.fr

Abstract

Interactive learning deals with the problem of learning and solving tasks using human instructions. It is common in human-robot interaction, tutoring systems, and in human-computer interfaces such as brain-computer ones. In most cases, learning these tasks is possible because the signals are predefined or an ad-hoc calibration procedure allows to map signals to specific meanings. In this paper, we address the problem of simultaneously solving a task under human feedback and learning the associated meanings of the feedback signals. This has important practical application since the user can start controlling a device from scratch, without the need of an expert to define the meaning of signals or carrying out a calibration phase. The paper proposes an algorithm that simultaneously assign meanings to signals while solving a sequential task under the assumption that both, human and machine, share the same a priori on the possible instruction meanings and the possible tasks. Furthermore, we show using synthetic and real EEG data from a brain-computer interface that taking into account the uncertainty of the task and the signal is necessary for the machine to actively plan how to solve the task efficiently.

1 INTRODUCTION

Interactive learning [1, 2] aims at developing systems that can learn by practical interaction with the user and finds applications in a wide range of fields such as human-robot interaction, tutoring systems or human-machine interfaces. This type of learning combines ideas of learning from demonstration [3], learning by exploration [4] and tutor feedback [5]. Under this approach the human teacher interacts with the machine and provides extra feedback or guidance.

Approaches have considered: extra reinforcement signals [6], action requests [7], disambiguation among actions [8], preferences among states [9], iterations between practice and user feedback sessions [10], and choosing actions that maximize the user feedback [11].

A usual assumption in such systems is that the learner and the teacher share a mutual understanding of the meaning of each others' signals, and in particular the learning agent is usually assumed to know how to interpret teaching instructions from the human. In practice, this problem is solved due to two simplifications. On the one hand, the range of accepted instructions is limited to those predefined by the system developer. This approach, commonly used in human-robot interaction, lacks flexibility and adaptation to user specificities and, consequently, may not be well accepted by non-experts users with different preferences. On the other hand, sometimes it is not enough to predefine the instruction sets and it is necessary to perform a calibration phase to map raw signals such as speech or brain activity to their meanings. This is usually done using an ad-hoc protocol to collect labeled samples of the user instruction signals. This process must be well controlled to ensure signals are associated to the true intended meaning of the user.

The previous engineering solution is needed due to the chicken egg nature of the problem. In order to teach a system a new skill, it needs to understand the human instructions. And, in order to understand this feedback, the system must have some interaction with the human (e.g. through a controlled task as done in the calibration process) to learn what the instructions mean. Few works have studied and developed interactive learning systems that can learn both the meaning of signals and the task simultaneously. In human-robot interaction Griffiths et al. [12] conducted an experiment with humans learning the meaning of unknown symbolic teaching signals. Lopes et al. [13] presented sequential task experiments considering symbolic teaching signals and requiring a bootstrap with known signals. Grizou et al. [14] extended their system for non-symbolic teaching signals while removing the need for bootstrapping with known signals. Which they later extended to non-

invasive brain-computer interfaces (BCIs), proposed an uncertainty measure on both the task and the signal model for efficient planning, and performed online experiments [15]. Also, for P300 spellers, Kindermans et al. have shown that it is possible to exploit the repetition of signals [16] together with prior information (language models, information from other subjects) [17] to calibrate the EEG decoder while using the speller. They exploit the particular fact that only one event out of six encodes a P300 potential in the speller paradigm. BCIs usually require user-dependent calibration and have to deal with the EEG brain signals non-stationarities. These facts, together with the poor signal-to-noise ratio of the EEG, make the EEG self-calibration one of the most challenging ones.

This paper aims at solving the general problem of developing machines that can execute a task from human instructions and simultaneously learn the communicative signals. Our approach is based on a discretization of the possible tasks into a finite number. Each task assigns different expected meanings to the instruction provided by the user. The machine solves the most likely task according to a pseudo-likelihood function computed using the corresponding task labels. The experimental results, both synthetic and based on real EEG data, show that in order to simultaneously recover the meanings and solve the task it is of paramount importance to take into account the uncertainty on both task and signal space.

Compared to the work of Grizou et al. [14, 15], we improve the algorithm formalism for both learning and planning, the robustness to noisy high-dimensional signals (e.g. EEG), and allow to seamlessly transition from task to task without changing the algorithm paradigm. Grizou et al. methods in [14] and [15] required a different set of equations for the first task than for the further ones where only a fixed classifier, common for all hypothesis, was used. Compared to the work of Kindermans et al. [16, 17], our approach is more general and do not need to rely on specific patterns in the signal occurrences, i.e. they exploit the fact that only one event out of six encodes a P300 potential in the speller paradigm. The setup considered in this paper can not guarantee a specific ratio of meanings between received feedback signals.

In the following section, we present the set of assumptions and algorithmic details of our system. Then we introduce the specificity of the uncertainty inherent to our problem using an intuitive example and present the details of our action selection method. Finally we present a set of simulated experiment showing that a) our action selection method is reliable and improve over other methods, b) our algorithm scale to the use of high dimensional signals coming from previously recorded brain signals, and c) by being operational from the first step, as opposed to calibration procedure, we can estimate the correct task as soon as sufficient evidence has been collected.

2 ALGORITHM

2.1 Problem definition

We consider interaction sessions where a machine can perform discrete actions from a set of available actions $a \in \mathcal{A}$ in an either discrete or continuous state space $s \in \mathcal{S}$. The user, that wants to achieve a task $\hat{\xi}$, is providing feedback to the machine using some specific signal e , represented as a feature vector. The task is sequential meaning it is completed by performing a sequence of actions. The machine ignores the task the user has in mind, as well as the actual meaning of each user's signal. Its objective is to simultaneously solve the task and learn a model for the user's signals. To achieve this, it has access to a sequence of triplets in the form $D_M = \{(s_i, a_i, e_i), i = 1, \dots, M\}$, where s_i , a_i and e_i represent, respectively, the state, action and instruction signals at time step i . The behavior of the machine is determined by the actions $a \in \mathcal{A}$ and the corresponding transition model $p(s' | s, a)$.

We make the following assumptions under this general paradigm. First, the system has access to a set of tasks ξ_1, \dots, ξ_T which includes the task the user wants to solve. We assume the instruction signals e have a finite and discrete number of meanings $l \in \{l_1, l_2, \dots, l_L\}$ which we call labels and this is known by the user and the machine. In this work we will consider two possible meanings for the signals: correct or incorrect; but more complex meanings could be used, such as guidance instructions (go up, go left, ...). We assume that given these labels, it is possible to compute a model that generates or classifies signals e into meanings l . The parameters of such a model will be denoted by θ and we assume this mapping between signal e and their label l to be fixed. However this mapping is unknown to the agent at start.

2.2 Estimating Tasks Likelihoods

We start by assuming we are provided a signal decoder $\hat{\theta}$ and relax this assumption later on. As mentioned in the introduction, knowing $\hat{\theta}$, we can compute the probability of each task ξ_t after observation of a signal e when performing action a in state s :

$$p(\xi_t | e, s, a, \hat{\theta}) \propto p(e | s, a, \hat{\theta}, \xi_t) p(\xi_t) \quad (1)$$

where $p(e | s, a, \hat{\theta}, \xi_t)$ needs to take into account the probability of each possible meaning l given the target ξ_t , the current state s and the action a executed by the machine:

$$p(e | s, a, \hat{\theta}, \xi_t) = \sum_{k \in 1, \dots, L} p(e | l = l_k, \hat{\theta}) p(l = l_k | s, a, \xi_t) \quad (2)$$

This process can be repeated recursively for several inter-

action steps i :

$$\begin{aligned}\mathcal{L}_i^{\xi_t} &= p(\xi_t | D_i^{\xi_t}, \hat{\theta}) \\ &\propto p(e_i | s_i, a_i, \hat{\theta}, \xi_t) p(\xi_t | D_{i-1}^{\xi_t}, \hat{\theta})\end{aligned}\quad (3)$$

with $p(\xi_t | D_0^{\xi_t}, \hat{\theta})$ being the prior at time 0 (before the experiment starts) for the task ξ_t , usually uniform over the task distribution.

We now relax the assumption we are given a model $\hat{\theta}$. The natural extension from the previous models is to compute the posterior distribution over the task and the model, $p(\xi, \theta | e, s, a)$. However, the resulting distribution does not have a close form solution even when linear Gaussian likelihoods are used due to the combination of mixtures for each possible task. Another alternative is to compute the θ and ξ that maximize the data likelihood. This is prone to fail in certain scenarios due to two reasons. First, it is common that different tasks share many labels (e.g. the policies to reach neighboring cells on a grid world are almost identical and, therefore, share most of the labels l) and results on large uncertainties in the task space that require multiple actions to be disambiguated. Second, if the signals are not well separated the meaning parameters θ of different tasks will not differ much.

For instance, under Gaussian assumptions for $p(e | l = l_k, \theta)$ and deterministic task labels $p(l = l_k | s, a, \xi)$, it is possible to integrate out θ to compute the marginal likelihood $p(D_M | \xi)$. The resulting likelihood depends only on the traces of each $p(e | l = l_k, \theta)$. Empirical results with synthetic and EEG data for a reaching task on a grid revealed that, when the distributions over e overlap, the traces were not enough to recover the most likely task and the corresponding meaning parameters.

To cope with these problems, we define the following pseudo-likelihood function:

$$P(D_M | \xi, \theta) \approx \prod_{i=1}^M p(e_i | s_i, a_i, \xi, \theta_{-i}) \quad (4)$$

$$= \prod_{i=1}^M \sum_{l_c} \sum_l p(e_i | l_c, \theta_{-i}) p(l_c | l, \theta_{-i}) p(l | s_i, a_i, \xi) \quad (5)$$

where l represents the meaning assigned by task ξ , action a_i and state s_i and l_c is the label corrected based on what we know about our classifier θ_{-i} for a given label l .

The pseudo-likelihood is built using a leave-one-out cross-validation strategy to evaluate the likelihood $p(e_i | s_i, a_i, \xi, \theta_{-i})$ of each signal based on the meaning parameters θ_{-i} learned for each task using all the other available signals. The use of θ_{-i} indicates we use a leave one out method. If we interpret $p(e_i | s_i, a_i, \xi, \theta_{-i})$ as a classifier, its predicted labels should match the ones provided by the task for different state-actions pairs. The rationale behind it is that for the correct task, the signals

and labels will be more coherent than for other tasks, which we measure as the predictive ability of a classifier trained on the signal-label pairs. Note that wrong tasks will assign wrong labels l to the signals e , therefore the learned models will have larger overlaps (see Figure 1c).

Each term of the pseudo-likelihood is computed from three terms. $p(l | s_i, a_i, \xi)$ represents the probability distributions of the meanings according to a task, the executed action and the current state. $p(l_c | l, \theta_{-i})$ encodes which label will be actually recovered by θ_{-i} . Intuitively, it models the quality of the model θ_{-i} . $p(e_i | l_c, \theta_{-i})$ is the likelihood of the signal given the meaning. The pseudo-likelihood is maximized in two steps. First, the maximum a posteriori estimate θ_{-i} of each task is computed. Then, the term $p(l_c | l, \theta_{-i})$ is approximated by the corresponding confusion matrix of the classifier based on θ_{-i} . It is the probability that the classifier itself is reliable in its prediction. Finally, the best task ξ should be the one that maximizes the pseudo-likelihood in Eq. 4.

2.3 Decision and Task Change

The machine must decide which task is the correct one. To do so, we define W^{ξ_t} the minimum of pairwise normalized likelihood between hypothesis ξ_t and each other hypothesis:

$$W^{\xi_t} = \min_{x \in 1, \dots, T \setminus \{t\}} \frac{P(D_M | \xi_t, \theta)}{P(D_M | \xi_t, \theta) + P(D_M | \xi_x, \theta)} \quad (6)$$

When it exists a t such that W^{ξ_t} exceeds a threshold $\beta \in [0.5, 1]$ we consider task ξ_t is the one taught by the user.

Once a task is identified with confidence, the robot executes it and prepares to receive instructions from the user to execute a new task. Assuming the user starts teaching a new task using the same kind of signals, we now have much more information about the signal model. Indeed, we are confident that the user was providing instructions related to the previously identified task; therefore we can infer the true labels of the past signals. We can now assign such labels to all hypothesis and by using the same algorithm we can start learning the new task faster as all hypothesis now share a common set of signal-label pairs. The meaning models for each hypothesis are still updated step after step until the new task is identified and labels reassigned.

3 PLANNING UNDER UNCERTAINTY

To solve our problem we need to identify simultaneously the task and how to interpret teaching signals. To do so the system has to explore regions that allow to disambiguate among hypothesis. There are several efficient model-based reinforcement learning exploration methods that add an exploration bonus for states that might provide more learn-

ing gains. Several theoretical results show that these approaches allow to learn tasks efficiently [18, 19]. We define an uncertainty measure and use model-based planning to select sequences of actions that guide the agent toward states that better identify the desired task.

In order to exemplify the specificity of our problem in terms of planning we present a simple experiment and compare the effect of different action selection strategies. In this scenario, the agent is in a T world with 7 states and can perform 4 actions (right, left, up, and down). The user wants the robot to reach the left edge (marked by G1) of the T, (see Figure 1 top). The agent knows the users wants it to go to one of the two edges (G1 or G2) but not which one. The agent will perform some actions, and the user will assess the correctness of each agent’s action by providing a two dimensional teaching signal. The agent does not know which signal means “correct” and which signal means “incorrect”. As there is two possible tasks, the agent will assign labels to every user’s signals according to each hypothesis. The result of the labeling process is displayed as colored dots (green for “correct” and red for “incorrect”) in Figure 1 (a, b, and c), where the left part corresponds to hypothesis 1 (G1) and the right part to hypothesis 2 (G2).

If the agent knew how to interpret the signal, i.e. which signal corresponds to correct or incorrect feedback, the optimal action to differentiate between the two hypothesis would be to perform right and left actions in the top part of the T. However in our problem the classifier is not given and the agent is building a different model for each hypothesis. As a results, we end up with two opposite interpretations of the user signal, which are both as valid (see Figure 1a) and do not allow to differentiate between hypothesis.

Considering that the agent does not know the signal to meaning classifier, a sensitive option is to select actions that allow to unequivocally identify the model. In our scenario taking only up and down actions in the trunk of the T leads to identical interpretation for each hypothesis (see Figure 1b). However this method do not allow to disambiguate between hypothesis and in most setting, such as the grid world we consider later, there is no state-action pair leading to unequivocal interpretations.

However performing all the four actions allow to disambiguate between hypothesis. As shown in Figure 1c, hypothesis 1 stands out by the nice coherence between the labels and the spacial organization of the data. This informs us that hypothesis 1 is the task the user has in mind and that feedback signals in the right and left part of the feature space means “correct” and “incorrect” respectively.

For our kind of problem the agent can not just try to differentiate hypothesis by finding state-action pair where expected feedback differs but should also collect data to build a good model or at least invalidate other models. Can we

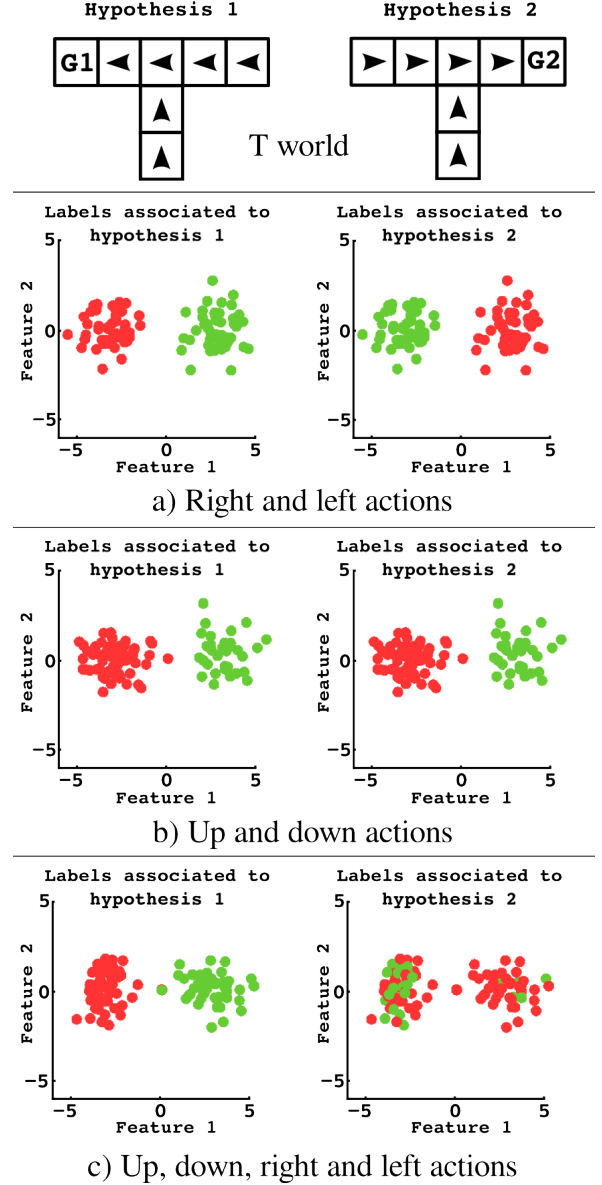


Figure 1: A “T world” scenario and the interpretation results for different planning strategies. The agent knows it should reach either of the two edges of the T world (marked with the letter G). The arrows represent the optimal policy. For each move the agent receives an unlabeled two dimensional teaching signal, corresponding to user’s assessments on the agent’s actions. The teacher’s goal is to have the agent reach G1. As the agent do not have access to this information, it interprets the signal according to each hypothesis (G1 and G2). a) shows the interpretation results if the agent only perform right and left actions in the top of the T world, b) shows the interpretation results when the agent only performs up and down actions in the trunk of the T, and c) shows the interpretation results for an agent performing all possible actions. Only the method c) allow to differentiate between hypothesis.

find a measure of uncertainty that account for both? Going back to Figure 1 (a and b), we understand that, to differentiate hypothesis in situation a) the best actions to perform are up and down in the T trunk while in situation b) the best actions to perform are right and left in the top part of the T. This corresponds to the uncertainty in the signal space. In the case of a) when going left both hypothesis agree that they will receive a signal in the right part of the feature space even if they disagree on its meaning. However for action down, both hypothesis agree they should receive a signal of meaning “incorrect” but disagree on the expected location of such signal in the feature space. In the case of b) when going up both hypothesis agree they will receive a signal in the right part of the feature space and agree on its meaning. However for action left, both hypothesis disagree about the meaning of the signal they should receive and as both share the same signal model they expect a signal in different locations of the feature space.

Estimating uncertainty in the signal space is in practice too costly as it requires to compute, for every state-action pair, the overlap between many continuous probability distributions weighted by their respective expected contribution. Following the discussion presented in previous section, we will rely on our pseudo-likelihood metric. As we cannot predict, neither control, the signal we will receive for a particular state-action, we will rely on our past history of signal and compute the expected joint probability based on previously experienced signals.

We note:

$$J^{\xi_t}(s, a, e) = \sum_{l_c} \sum_l p(e|l_c, \theta) p(l_c|l, \theta) p(l|s, a, \xi_t)$$

which is Eq. 5 for only one new expected observation e , so the product over iterations disappears. And $J^{\xi}(s, a, e)$ the vector $[J^{\xi_1}(s, a, e), \dots, J^{\xi_T}(s, a, e)]$.

The uncertainty of one state-action pair given a signal e is computed as the weighted variance of the joint probability predictions with weights $W^{\xi} = [W^{\xi_1}, \dots, W^{\xi_T}]$ (see Eq. 6):

$$U(s, a|e) = \text{weightedVariance}(J^{\xi}(s, a, e), W^{\xi}) \quad (7)$$

The uncertainty for a state-action pair is given by:

$$U(s, a) = \int_e U(s, a|e) p(e) de \quad (8)$$

which we approximate by summing values of $U(s, a|e)$ for different signals e :

$$U(s, a) \approx \sum_e U(s, a|e) p(e) \quad (9)$$

with $p(e)$ assumed uniform.

Our measure of global uncertainty $U(s, a)$ will be higher when, for a given state-action there is a high incongruity of expectation between each hypothesis and according to each hypothesis current probability.

This measure is then used as a classical exploration bonus method. We will switch to a pure exploitation of the task after reaching the desired confidence level.

Interestingly this approach generalizes over other active sampling method [7], if the classifier is known, equation 7 reduces to the one presented in [13] and is no longer dependent on signal e . As our uncertainty function combines uncertainty on both signal and task space, when the former is known, the latter becomes the sole source of ambiguity.

4 METHOD

In the subsequent analysis, we assume that a trainer provides feedback for the actions taken by a learner. Specifically, we consider the user is delivering signals that can be mapped into binary feedback: correct c and incorrect w .

4.1 World and Task

We consider a 5x5 grid world, where an agent can perform five different discrete actions: move up, down, left, right, or a “no move” action. The user goal is to teach the agent to reach one (unknown to the agent) of the 25 discrete positions which represent the set of possible tasks. We thus consider that the agent has access to 25 different task hypothesis (one with goal location at each of the cells). We use *Markov Decision Processes* (MDP) to represent the problem [4]. From a given task ξ , represented as a reward function, we can compute the corresponding policy π^{ξ} using, for instance, Value Iteration [4]. The policies allow us to interpret the teaching signals with respect to the interaction protocol defined. For the current work we will consider the user is providing feedback on the agent action. We define $p(l|s, a, \xi)$ as:

$$p(l|s, a, \xi) = \begin{cases} 1 - \alpha & \text{if } a = \arg\max_a \pi^{\xi}(s, a) \\ \alpha & \text{otherwise} \end{cases}$$

with α modeling the expected error rate of the user.

4.2 Signal properties and classifier

We aim at applying this algorithm to error-related potentials (ErrPs) for EEG-based BCI applications. These signals are generated in the user’s brain after s/he assesses actions performed by an external agent [20], where correct and erroneous assessments will elicit different brain signals. Past approaches have already demonstrated that these signals can be classified online with accuracies of around 80% and translated into binary feedback, thanks to a prior calibration session that lasts for 30-40 minutes [20, 21].

Following the literature [22], we will model the signals using independent multivariate normal distributions for each class, $\mathcal{N}(\mu_c, \Sigma_c), \mathcal{N}(\mu_w, \Sigma_w)$. With θ the set of parameters $\{\mu_c, \Sigma_c, \mu_w, \Sigma_w\}$. Given the high dimensionality of the problem we will also need to regularize. For this we apply shrinkage to the covariance matrix ($\lambda = 0.5$) and compute the value of the marginal pdf function using a non-informative (Jeffreys) prior [[23], p88]:

$$p(e|l, \theta) = t_{n-d}(e|\mu_l, \frac{\Sigma_l(n+1)}{n(n-d)}) \quad (10)$$

where θ represents the ML estimates (mean μ_l and covariance Σ_l for each class l) required to estimate the marginal under the Jeffreys prior, n is the number of signals, and d is the dimensionality of a signal feature vector.

4.3 Task Achievement

A task is considered completed when the confidence level β as been reached for this task and the agent is located at the task associated goal state. If the state is the one intended by the user it is a success. Whatever the success or failure of the first task, the user selects a new goal state randomly, the agent resets task likelihoods, propagates the believed labels, and teaching starts again. At no point the agent has access to a measure of its performance, it can only refer to the unlabeled feedback signals from the user.

4.4 Evaluation scenarios

Two different evaluation scenarios were tested with two different types of signals: artificial datasets, and real ErrP datasets recorded from previous experiments [21].

Artificial datasets The goal of this evaluation was to analyze the feasibility of learning a task from scratch in a 5x5 grid world. The artificial dataset was composed of two classes, with 1000 examples per class. Each example was generated by sampling from a normal distribution with a covariance matrix of diagonal 1 and mean selected randomly. The datasets were generated while varying two factors: (i) the dimensionality of the data, where 2, 5, 10 and 30 features were tested; and (ii) the quality of the dataset, measured in terms of the ten-fold accuracy the classifier would obtain.

Once the datasets were generated, two different evaluations were performed: (i) the goodness of our proposed planning strategy versus a) random action selection, b) greedy action selection, and c) a task-only uncertainty based method; (ii) the time required by the agent to learn the first task (i.e. to reach the first target), and (iii) the number of tasks that can be learned in 500 iterations.

EEG datasets Once the algorithm was evaluated with artificial datasets, we tested the feasibility of the proposed

self-calibration approach using real ErrP datasets. The objective of this analysis is to study the scalability of our method to EEG data, which may have different properties than our artificial dataset.

The EEG data were recorded in a previous study [21] where participants monitored on a screen the execution of a task where a virtual device had to reach a given goal. The motion of the device could be correct (towards the goal) or erroneous (away from the goal). The subjects were asked to mentally assess the device movements as erroneous or non-erroneous. The EEG signals were recorded with a gTec system with 32 electrodes distributed according to an extended 10/20 international system with the ground on FPz and the reference on the left earlobe. The ErrP features were extracted from two fronto-central channels (FCz and Cz) within a time window of [200, 700] ms (being 0 ms the action onset of the agent) and downsampled to 32 Hz. This led to a vector of 34 features.

Comparison with calibration methods In order to show the benefit of learning without explicit calibration, we compare our method with the standard supervised BCI calibration procedure. In this calibration procedure, which can last for up to 40 minutes, the experimenter needs to record enough data from the user from several offline runs, where the user is not controlling the agent but just passively assessing its actions. Following the literature on ErrPs [20, 21] our training data will consist of 80 percent of positive examples (associated to a correct feedback) and 20 percent of negative examples (associated to an incorrect feedback). Our proposed algorithm is compared with different (but standard) sizes of calibration datasets: 200, 300 and 400 examples.

4.5 Settings

We used $\alpha = 0.1$, $\beta = 0.9$. For dataset of dimension d , we started computing likelihoods after $d+10$ steps as equation 10 requires at least $d+1$ samples and to allow for cross validation. For the planning (Eq. 9) we selected randomly 20 signals from D_M .

5 RESULTS

We present most of the results in terms of the quality of the dataset, measured as the ten-fold classification accuracy that a calibrated signal classifier would obtain. Each simulation was run 100 times using different sampled datasets, and their associated box plots were computed. For each boxplot, colored bars show the interquartile range (between 25th and 75th percentile), and the median and the mean are marked as a horizontal line and a colored dot respectively. Additionally, the two “whiskers” show the 5th and 95th percentiles, black crosses are outliers.

5.1 Artificial Datasets

The first objective is to study the impact of the exploration approach proposed in Section 3. The second is to evaluate performances and robustness with respect to the dimension and the quality of each dataset.

Planning Methods Figure 2 compares the number of steps (with maximum values of 500 steps) needed to identify the first task when learning from scratch with different planning methods. Following the most probable task (i.e. going greedy) does not allow the system to explore sufficiently. On the contrary, our proposed planning method leads the system towards regions that maximize disambiguation among hypotheses. Furthermore, it also performs better than assessing uncertainty on the task space only. Given these results, the remainder of this section will only consider our planning method.

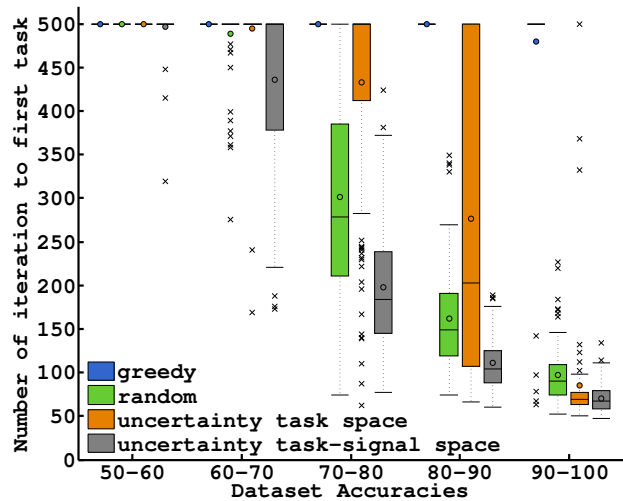


Figure 2: Number of steps to complete first task, comparison of different exploration methods with 30 dimensional artificial data. When learning from scratch, planning upon uncertainty in both task and signal space performs better than relying only on task uncertainty. Greedy action selection rarely disambiguates between hypothesis.

As depicted in Figure 1, the system needs to collect two types of information, some about the true underlying model (Fig. 1b) and some to differentiate between hypotheses (Fig. 1a). The properties of the grid world make the random strategy quite efficient at collecting those two types of information. The differences between planning methods should be more evident when navigating a complex maze since our method allows to plan in order to collect the type of information we need. Studying how different world properties affect the learning efficiency is part of our future work. Also, we note that all planning methods were switched to pure exploitation (greedy) once the confidence level was reached. Therefore the performance in Figure 2

compares the ability of the different methods to discriminate between different task hypotheses, not their ability to solve the task itself.

Dimensionality Figure 3 compares the number of steps (with maximum values of 500 steps) needed to identify the first task when learning from scratch with different dimensionality of datasets. The convergence speed is only slightly affected by the features dimensionality. On the other hand, the dataset quality (measured in terms of its associated ten-fold accuracy) is the main cause of performances decay. Furthermore, for those datasets with accuracies between 50% and 60%, the system is not able to identify a task with confidence after 500 steps.

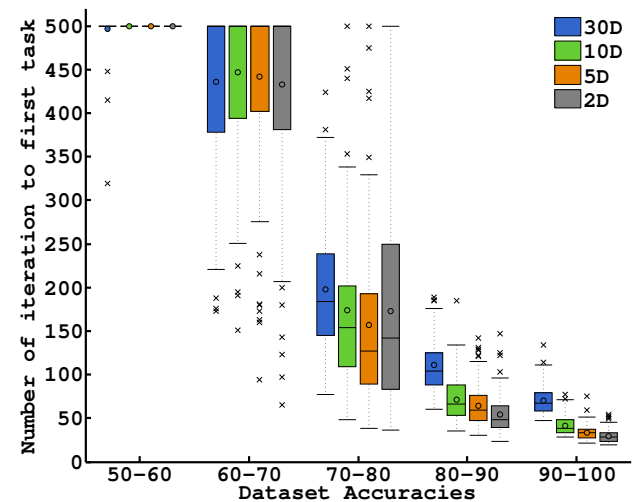


Figure 3: Number of steps to complete first task using artificial data. Under 60 percent accuracy, the confidence threshold cannot be reached in 500 steps. The dataset qualities, more than their dimensionality, impact the learning time.

Once one task is completed, a new one is selected randomly. Figure 4 compares the number of tasks that can be achieved in 500 steps. As expected, the lower the quality of the data, the less number of task can be completed. With dataset accuracies higher than 90% we can achieve more than 30 tasks on average.

An important aspect of the proposed learning approach was that the first task learned was always the correct one. We reported only 9 erroneous estimations across all simulated experiments (5 in the 70-80 group and 4 in the 80-90 group).

5.2 EEG datasets and comparison with calibration method

Example Figure 5 shows one particular run of 500 steps comparing our self-calibration method with a calibration

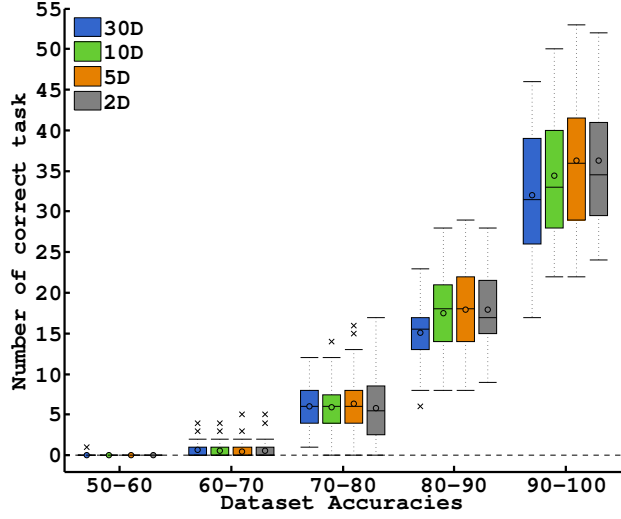


Figure 4: Number of tasks correctly achieved in 500 steps, artificial data. Quality of dataset impacts the number of task identified in 500 steps as more evidence should be collected to reach the confidence threshold.

procedure of 400 steps. The two independent runs use as real EEG dataset with 80% ten-fold classification accuracy. As our algorithm is operational from the first step, it can estimate the real task when sufficient evidence has been collected. On the other hand, a calibration approach collects signal-label pairs for a fixed number of steps and use the resulting classifier without updating it. This provokes that, during the calibration phase, no tasks can be learned, substantially delaying the user's online operation.

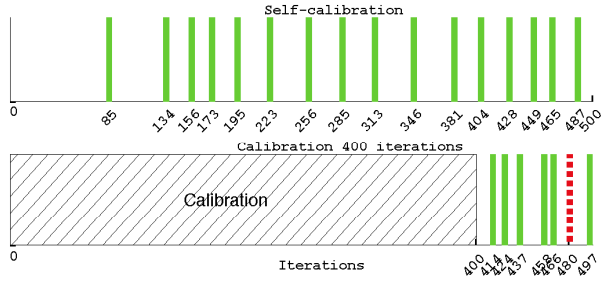


Figure 5: Time-line of one run from EEG dataset of 80 percent ten-fold classification accuracy, self-calibration (top) versus 400 steps calibration (bottom). Green (filled) and red (dashed) bars represents respectively correct and incorrect task achievement. The proposed self-calibration method allow to reach a first task faster than would take a calibration procedure.

Figure 6 shows the evolution of classification rate between our self-calibration method with a calibration procedure of 400 steps. As our method assigns different labels to each new teaching signal, the resulting classifiers have different performances, which help identifying the correct task.

Once a task is identified (e.g. step 85 and 134), the corresponding labels are taken as ground truth, and all classifiers will have the same accuracies. As the agent starts exploring again to estimate the new tasks, all the classifiers except the true one will start to have worse accuracies again.

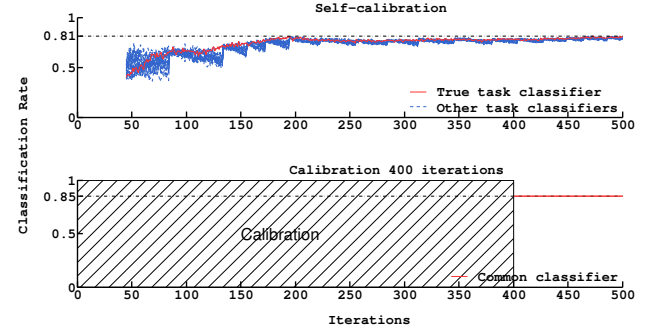


Figure 6: Evolution of classification rate of one run from EEG data, self-calibration (top) versus 400 steps calibration (bottom). On top, the red line represents the classifier corresponding to the successive tasks taught by the user, the dashed blue lines represent all others tasks. Our method updates classifiers every steps.

Time to first task Figure 7 shows the results per group of dataset. Our algorithm allows to complete the first task without errors and in a fair amount of iteration. For our method, the learning time is strongly correlated with the dataset quality. However calibration methods, which do not update their classifier once calibrated, identify more tasks incorrectly.

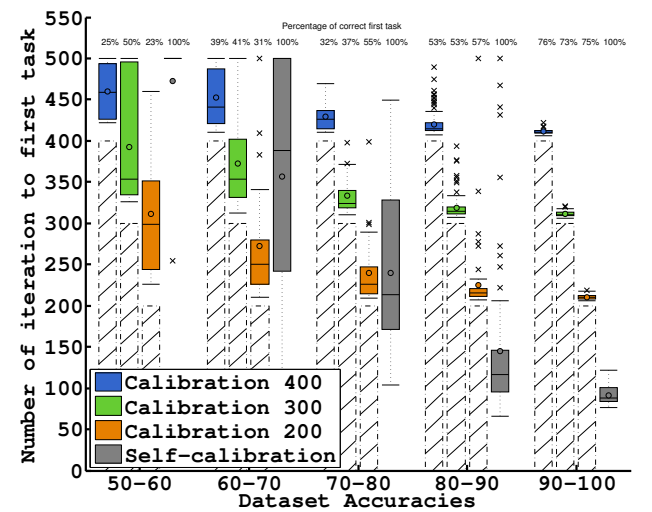


Figure 7: Number of steps to complete first task with EEG data. The method scale well to EEG data. Contrary to the standard calibration approaches, we do not make mistakes with low quality datasets.

Cumulative performances Figure 8 compares the number of tasks that can be achieved in 500 steps. With 90% and more dataset quality we can achieve about 20 tasks on average. The results are consistent with artificial dataset analysis.

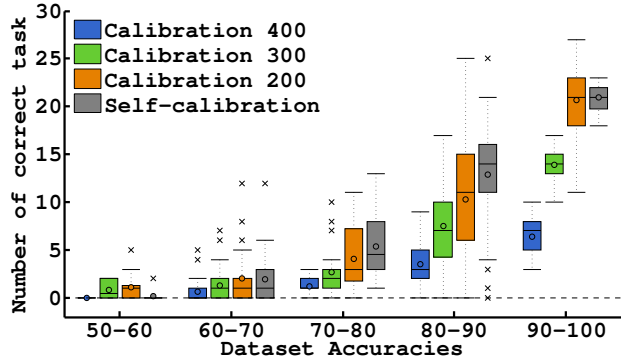


Figure 8: Number of task correctly achieved in 500 steps with EEG data. Calibration methods can not complete a significant number of task as most of the time is spent on calibration.

The calibration methods can not complete many task as a significant amount of iteration was used for calibrating the system. A calibration of 200 steps makes as many good estimation than our method, but it also makes many wrong estimation, see Figure 9. For calibration methods, the less time spent on calibration, the poorer the classifier which implies more mistakes.

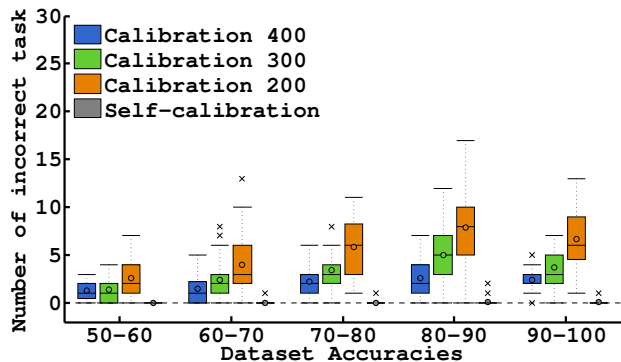


Figure 9: Number of task incorrectly achieved in 500 steps with EEG data. Calibration methods, which do not update their models once calibrated, make more errors.

6 CONCLUSION

In this paper we have shown that, given a limited number of possible tasks, it is possible to solve sequential tasks using human feedback without defining a map between feedback signals and their meaning beforehand. The proposed algorithm optimizes a pseudo-likelihood function and performs active planing according to the uncertainty in the task

and meaning spaces. Indeed, taking into account this uncertainty is crucial to solve the task efficiently and to recover the actual meanings. This combination allows: a) a human to start interacting with a system without calibration; b) to automatically adapt calibration time to the user needs which can even outperform fixed calibration procedures; c) to adapt to the uncertainty of the information source from scratch. We showed the applicability of the approach to brain-machine interfaces based on error potentials which could work out of the box without calibration, a long-desired property of this type of systems.

A number of open questions remain to be addressed:

- How the task properties (symmetries, size, ...) affect the learning properties?
- How to leverage from the finite set of hypothesis constraint? A potential avenue is to use a combination of particle filter and regularization on the task space.
- In real-world applications, users are usually told how to interact with machines. Do people want to have an open-ended choice about what signal to use? Would they be more efficient? When is it better to use a calibration procedure?
- Only prerecorded datasets have been used. However, signals may change during the learning. For instance, people can try to adapt themselves to a robot if they believe the latter is not understanding properly. Or, brain signals are sensitive to the protocol, the duration of the experiment or even the percentage of errors made by the agent [20]. To which extend the behavior of our agent changes the properties of the teaching signal? Can we adapt to such changes online?

Finally, while we only considered correct/incorrect labels, in other works we have considered the use of guidance instructions (go up, go left, ...) in human-robot interaction scenario [14]. But increasing the set of possible labels logically requires collecting more examples to obtain a good enough representation of the different signals. Hence, for BCI domains, it is reasonable to keep a limited number of labels.

Acknowledgments

We thank the anonymous reviewers for their helpful comments. Work partially supported by INRIA, Conseil Régional d'Aquitaine, the 3rd Hand Project (funded under 7th FWP), and the ERC grant EXPLORERS 24007; and from the Spanish Ministry via DPI2011-25892 (L.M.), and DGA-FSE grupo T04.

References

- [1] M. N. Niculescu and M. J. Mataric, "Natural methods for robot task learning: Instructive demonstrations, generalization and practice," in *Conference on Autonomous agents and multiagent systems*, pp. 241–248, ACM, 2003.
- [2] C. Breazeal, A. Brooks, J. Gray, G. Hoffman, C. Kidd, H. Lee, J. Lieberman, A. Lockerd, and D. Chilongo, "Tutelage and collaboration for humanoid robots," *International Journal of Humanoid Robotics*, vol. 1, no. 02, pp. 315–348, 2004.
- [3] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.
- [4] R. Sutton and A. Barto, *Reinforcement learning: An introduction*, vol. 28. Cambridge Univ Press, 1998.
- [5] F. Kaplan, P.-Y. Oudeyer, E. Kubinyi, and A. Miklósi, "Robotic clicker training," *Robotics and Autonomous Systems*, 2002.
- [6] A. L. Thomaz and C. Breazeal, "Teachable robots: Understanding human teaching behavior to build more effective robot learners," *Artificial Intelligence*, vol. 172, no. 6, pp. 716–737, 2008.
- [7] M. Lopes, F. Melo, and L. Montesano, "Active learning for reward estimation in inverse reinforcement learning," in *Machine Learning and Knowledge Discovery in Databases*, pp. 31–46, Springer, 2009.
- [8] S. Chernova and M. Veloso, "Interactive policy learning through confidence-based autonomy," *Journal of Artificial Intelligence Research*, vol. 34, no. 1, p. 1, 2009.
- [9] M. Mason and M. Lopes, "Robot self-initiative and personalization by learning through repeated interactions," in *Human-Robot Interaction (HRI), 2011 6th ACM/IEEE International Conference on*, pp. 433–440, IEEE, 2011.
- [10] K. Judah, S. Roy, A. Fern, and T. G. Dietterich, "Reinforcement learning via practice and critique advice," in *AAAI*, 2010.
- [11] W. B. Knox and P. Stone, "Interactively shaping agents via human reinforcement: The tamer framework," in *International conference on Knowledge capture*, pp. 9–16, ACM, 2009.
- [12] S. Griffiths, S. Nolfi, G. Morlino, L. Schillingmann, S. Kuehnel, K. Rohlfing, and B. Wrede, "Bottom-up learning of feedback in a categorization task," in *Development and Learning and Epigenetic Robotics (ICDL), 2012*, pp. 1–6, IEEE, 2012.
- [13] M. Lopes, T. Cederborg, and P.-Y. Oudeyer, "Simultaneous acquisition of task and feedback models," in *IEEE - International Conference on Development and Learning (ICDL'11)*, 2011.
- [14] J. Grizou, M. Lopes, and P.-Y. Oudeyer, "Robot Learning Simultaneously a Task and How to Interpret Human Instructions," in *Conference on Development and Learning and on Epigenetic Robotics (ICDL-EpiRob)*, (Osaka, Japan), 2013.
- [15] J. Grizou, I. Iturrate, L. Montesano, P.-Y. Oudeyer, and M. Lopes, "Calibration-Free BCI Based Control," in *AAAI Conference on Artificial Intelligence*, (Quebec, Canada), 2014.
- [16] P.-J. Kindermans, D. Verstraeten, and B. Schrauwen, "A bayesian model for exploiting application constraints to enable unsupervised training of a P300-based BCI," *PloS one*, vol. 7, p. e33758, Jan. 2012.
- [17] P.-J. Kindermans, M. Tangermann, K.-R. Müller, and B. Schrauwen, "Integrating dynamic stopping, transfer learning and language models in an adaptive zero-training erp speller," *Journal of neural engineering*, vol. 11, no. 3, p. 035005, 2014.
- [18] R. Brafman and M. Tennenholtz, "R-max-a general polynomial time algorithm for near-optimal reinforcement learning," *Journal of Machine Learning Research*, vol. 3, 2003.
- [19] J. Z. Kolter and A. Y. Ng, "Near-bayesian exploration in polynomial time," in *International Conference on Machine Learning*, ACM, 2009.
- [20] R. Chavarriaga and J. Millán, "Learning from EEG error-related potentials in noninvasive brain-computer interfaces," *IEEE Trans Neural Syst Rehabil Eng*, vol. 18, no. 4, 2010.
- [21] I. Iturrate, L. Montesano, and J. Minguez, "Task-dependent signal variations in eeg error-related potentials for brain-computer interfaces," *Journal of neural engineering*, vol. 10, no. 2, p. 026024, 2013.
- [22] B. Blankertz, S. Lemm, M. Treder, S. Haufe, and M. KR, "Single-trial analysis and classification of ERP components: A tutorial," *Neuroimage*, 2010.
- [23] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian data analysis*. CRC press, 2003.

Batch-Mode Active Learning via Error Bound Minimization

Quanquan Gu

Dept. of Computer Science
University of Illinois
Urbana, IL 61801
qgu3@illinois.edu

Tong Zhang

Department of Statistics
Rutgers University
Piscataway, NJ 08854
tzhang@stat.rutgers.edu

Jiawei Han

Dept. of Computer Science
University of Illinois
Urbana, IL 61801
hanj@cs.uiuc.edu

Abstract

Active learning has been proven to be quite effective in reducing the human labeling efforts by actively selecting the most informative examples to label. In this paper, we present a batch-mode active learning method based on logistic regression. Our key motivation is an out-of-sample bound on the estimation error of class distribution in logistic regression conditioned on any fixed training sample. It is different from a typical PAC-style passive learning error bound, that relies on the i.i.d. assumption of example-label pairs. In addition, it does not contain the class labels of the training sample. Therefore, it can be immediately used to design an active learning algorithm by minimizing this bound iteratively. We also discuss the connections between the proposed method and some existing active learning approaches. Experiments on benchmark UCI datasets and text datasets demonstrate that the proposed method outperforms the state-of-the-art active learning methods significantly.

1 INTRODUCTION

In a typical supervised learning problem, one often requires sufficient labeled data to train an accurate classifier, whereas the labeling process may be expensive and time consuming. This motivates *Active Learning* [11], which has been proven to be effective in reducing the human labeling efforts by actively selecting the most informative examples for labeling. The goal of active learning is to learn a classifier which accurately predicts the labels of new examples, while requesting as few labels as possible.

In the past decades, many active learning methods have been proposed. Depending on the label query strategy, active learning can be roughly categorized into fully sequential active learning [13, 27, 30, 5, 24, 7, 22], batch-

mode active learning [20, 17, 21] and one-shot active learning [29, 15, 16, 14]. Fully sequential active learning algorithms select only one example to query its label at one time, and update the classifier. In contrast, batch-mode active learning algorithms select multiple examples at one time. It is more efficient since the classifier is trained fewer times. More importantly, it is able to take into account the information overlap among the multiple examples. Both fully sequential and batch-model active learning are adaptive, as in the query process, the newly labeled data in an earlier iteration can be used to guide the selection of unlabeled data in a latter iteration (e.g., by updating the classifier). In contrast, one-shot active learning is non-adaptive. In this paper, we consider batch-mode active learning, because it is more general than the other two query strategies both in theory and practice. It can be directly adapted to fully sequential and one-shot active learning, by simply setting the batch-size to one or to a sufficient large number.

On the other hand, the most widely used criteria for active learning include but not limited to uncertainty sampling [27, 21], query by committee [13], mutual information [24, 16], experimental design [29, 3], and expected error minimization [15, 14]. Besides these practical algorithms mentioned above, there are also several theoretical studies [5, 12, 7, 18, 1], which provide bounds on the label complexity. The method we are going to propose belongs to the family of expected error minimization. The main advantage of the methods in this family is that the criteria are minimizing certain kind of error bounds, which directly relate the label selection procedure with the prediction error. As a result, we are particularly interested in designing such kind of active learning algorithm.

With the above motivation, we present a batch-mode active learning method, which is based on the well-known statistical model of logistic regression [19]. One advantage of logistic regression is that it has an inherent model assumption and thus it is amenable to theoretical analysis. Furthermore, it is in nature a classification model and consequently more suitable for active learning towards classification. We perform a finite sample analysis on the logistic regression

and derive an error bound on the class distribution conditioned on any fixed training sample. This bound is essential because it is different from a typical PAC-style error bound for model-free passive learning [8], that relies on the i.i.d. assumption of the example-class pairs. In contrast, our derived bound allows the training examples to be dependent, which meets the scenario of active learning. Furthermore, the derived error bound does not contain the class labels of the training sample, which allows us to do minimization by choosing training examples without knowing their labels. We propose an active learning criterion to select the examples by minimizing this upper bound iteratively. The resulting method is a combinatorial optimization problem, which is relaxed and solved approximately by projected gradient descent. It is worth noting that the derivation approach we proposed is quite general and is applicable to other generalized linear models beyond logistic regression.

As we mentioned before, although we mainly study batch-mode active learning in this paper, our proposed method supports fully sequential and one-shot active learning as well. Furthermore, unlike many active learning methods [5, 12, 7], which rely on sampling the hypothesis space, our method is deterministic and easy to implement. Extensive experiments on UCI datasets and text datasets show that the proposed method significantly outperforms the state-of-the-art active learning methods.

The remainder of this paper is organized as follows. In Section 2, we analyze the logistic regression, and derive a finite sample error bound on its response distribution. In Section 3, we present an active learning criterion based on minimizing the derived error bound, followed by its optimization algorithm. We discuss some related methods in Section 4. The experiments are demonstrated in Section 5. Finally, we draw conclusions and point out the future work in Section 6.

2 FINITE SAMPLE ANALYSIS OF LOGISTIC REGRESSION

In this section, to keep this paper self-contained, we first briefly review logistic regression [19]. Then we derive an estimation error bound for the conditional class distribution based on finite-sample analysis. It is among the main contributions of this paper, and is the theoretical underpinning of the active learning approach proposed in the next section.

2.1 NOTATION

Throughout this paper, we will use lower case letters to denote scalars, lower case bold letters to denote vectors, upper case letters to denote the elements of a matrix or a set, and bold-face upper case letters to denote matrices. \mathbf{I} is an identity matrix with an appropriate size. We

use superscript $^\top$ to denote the transpose of a vector or a matrix. The ℓ_2 -norm of a vector $\mathbf{x} \in \mathbb{R}^d$ is defined as $\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^d x_i^2}$. The spectral norm of a matrix \mathbf{A} is defined as $\|\mathbf{A}\|_2 = \max_{\|\mathbf{x}\|_2=1} \|\mathbf{A}\mathbf{x}\|_2$. In particular, for a squared matrix $\mathbf{A} \in \mathbb{R}^{d \times d}$, we denote its maximum eigenvalue by $\lambda_{\max}(\mathbf{A})$, and its minimum eigenvalue by $\lambda_{\min}(\mathbf{A})$. We use $[n]$ to denote the index set $\{1, 2, \dots, n\}$. Given a matrix $\mathbf{X} \in \mathbb{R}^{d \times n}$, $\mathbf{X}_{\mathcal{L}}$ denotes a submatrix of \mathbf{X} , which consists of the columns of \mathbf{X} indexed by $\mathcal{L} \subset [n]$. \mathbf{x}_i denotes the i -th column of \mathbf{X} . And for a symmetric matrix $\mathbf{D} \in \mathbb{R}^{n \times n}$, $\mathbf{D}_{\mathcal{L}\mathcal{L}}$ denotes a submatrix of \mathbf{D} , which contains the rows and columns indexed by \mathcal{L} .

2.2 LOGISTIC REGRESSION

Let us consider the binary classification case for simplicity. Given a sample set $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{\pm 1\}$, to have a simpler derivation without considering the bias term θ , one often augments each example with an additional dimension: $\mathbf{x}^\top \leftarrow [\mathbf{x}^\top; 1]$ and $\mathbf{w}^\top \leftarrow [\mathbf{w}^\top; \theta]$. In logistic regression [19], the conditional class probability $\Pr(y|\mathbf{x})$ is given by

$$\Pr(y|\mathbf{x}; \mathbf{w}) = \sigma(y\mathbf{w}^\top \mathbf{x}),$$

where $\sigma(a)$ is the logistic sigmoid function, i.e., $\sigma(a) = 1/(1 + \exp(-a))$. Note that $\sigma(a)$ is a concave function when $a > 0$.

To avoid over-fitting, we place a prior on \mathbf{w} in the form of a zero-mean Gaussian distribution with isotropic covariance, i.e., $\mathcal{N}(\mathbf{0}, 1/\lambda \mathbf{I})$, and seek a \mathbf{w} which maximizes the log-likelihood of the posterior distribution given the training data S , which is equivalent to

$$\hat{\mathbf{w}} = \arg \min_{\mathbf{w}} \lambda \|\mathbf{w}\|_2^2 - \frac{1}{n} \sum_{i=1}^n \log \sigma(y_i \mathbf{w}^\top \mathbf{x}_i), \quad (1)$$

where λ is a positive regularization parameter. Eq. (1) is also known as penalized logistic regression, or more precisely, ℓ_2 -regularized Logistic regression. It is worth noting that although logistic regression is called “regression”, it is in nature a classification model, because its response is binary and it directly estimates the conditional class probability given the data. This is also the reason that we deem that deriving an active learning algorithm based on logistic regression is more natural and effective for classification than inventing one from the real regression models [29, 14].

2.3 ERROR BOUNDS FOR LOGISTIC REGRESSION

In the following, we will analyze ℓ_2 -regularized logistic regression reviewed above. First of all, we assume that there exists an unknown true parameter $\mathbf{w}_* \in \mathbb{R}^d$, by which the class label of an example is generated as follows

$$\Pr(y|\mathbf{x}) = \sigma(y\mathbf{w}_*^\top \mathbf{x}). \quad (2)$$

where $\|\mathbf{w}_*\|_2 \leq R$ for some $R > 0$. This is our model assumption. All the theoretical results we are going to present are built up on this assumption.

Without loss of generality, we assume $\|\mathbf{x}_i\|_2 \leq 1$ for $\forall i$. Then it is easy to verify that

$$\left\| \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i \mathbf{x}_i^\top \right\|_2 \leq 1.$$

The following theorem provides a bound on the estimation error of $\hat{\mathbf{w}}$, which is central in our theoretical results. The detailed proofs can be found in the supplementary material.

Theorem 1. For any fixed sample $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, where y_i follows the conditional distribution as in Eq. (2), and $\|\mathbf{x}_i\|_2 \leq 1$ for $\forall i$. $\hat{\mathbf{w}}$ is the estimated weight vector by logistic regression on S , then the estimation error of $\hat{\mathbf{w}}$ is upper bounded as

$$\begin{aligned} & \mathbb{E}_{Y|X} [\|\hat{\mathbf{w}} - \mathbf{w}_*\|_2] \\ & \leq C_1 \lambda_{\max} \left(\left(\lambda \mathbf{I} + \frac{1}{n} \mathbf{X} \mathbf{D} \mathbf{X}^\top \right)^{-1} \right), \end{aligned}$$

where $\mathbb{E}_{Y|X}$ is the shorthand for $\mathbb{E}_{y_1, \dots, y_n | \mathbf{x}_1, \dots, \mathbf{x}_n}$, $C_1 = 1 + 2\lambda R$, $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]$, \mathbf{D} is a diagonal matrix with diagonal elements defined as follows

$$D_{ii} = (1 - \sigma(\mathbf{w}_*^\top \mathbf{x}_i)) \sigma(\mathbf{w}_*^\top \mathbf{x}_i). \quad (3)$$

Proof. (Sketch of proof): We use a similar technique adopted in [25]. Define $f(\mathbf{w}) = \lambda \|\mathbf{w}\|_2^2 - 1/n \sum_{i=1}^n \log \sigma(y_i \mathbf{w}^\top \mathbf{x}_i)$. Let $\hat{\mathbf{w}} = \arg \min_{\mathbf{w}} f(\mathbf{w})$.

Define $g(\Delta)$ as follows

$$\begin{aligned} g(\Delta) = & \mathbb{E}_{Y|X} [\lambda \|\mathbf{w}_* + \Delta\|_2^2 - \frac{1}{n} \sum_{i=1}^n \log \sigma(y_i (\mathbf{w}_* + \Delta)^\top \mathbf{x}_i) \\ & - \lambda \|\mathbf{w}_*\|_2^2 + \frac{1}{n} \sum_{i=1}^n \log \sigma(y_i \mathbf{w}_*^\top \mathbf{x}_i)], \end{aligned}$$

where $\Delta = \mathbf{w} - \mathbf{w}_*$. It is easy to verify that $g(0) = 0$. Using the optimality of $\hat{\mathbf{w}}$, we have $f(\hat{\mathbf{w}}) \leq f(\mathbf{w}_*)$, yielding

$$\begin{aligned} & \lambda \|\hat{\mathbf{w}}\|_2^2 - \frac{1}{n} \sum_{i=1}^n \log \sigma(y_i \hat{\mathbf{w}}^\top \mathbf{x}_i) \\ & \leq \lambda \|\mathbf{w}_*\|_2^2 - \frac{1}{n} \sum_{i=1}^n \log \sigma(y_i \mathbf{w}_*^\top \mathbf{x}_i). \end{aligned}$$

Therefore, we have $g(\hat{\Delta}) \leq 0$ with $\hat{\Delta} = \hat{\mathbf{w}} - \mathbf{w}_*$. Suppose that we show for some radius $B > 0$, and for $\Delta \in \mathbb{R}^d$ with $\|\Delta\|_2 = B$, we have $g(\Delta) > 0$. We then can claim that $\|\hat{\Delta}\|_2 \leq B$. We prove it by contradiction: If $\hat{\Delta}$ lies outside the ball of radius B , then by convexity of $g(\cdot)$, we have

$$g(t\hat{\Delta} + (1-t)0) \leq tg(\hat{\Delta}) + (1-t)g(0) \leq 0,$$

for some appropriately chosen $t \in (0, 1)$ such that $t\hat{\Delta} + (1-t)0$ lies on the boundary of the ball. This is contradict with the fact that $g(t\hat{\Delta} + (1-t)0) > 0$.

By some calculations, we have

$$g(\Delta) \geq C_{\min} B^2 - B - 2\lambda R B + \lambda B^2,$$

where $C_{\min} = \lambda_{\min}(1/n \sum_{i=1}^n \sigma(\mathbf{w}_*^\top \mathbf{x}_i)(1 - \sigma(\mathbf{w}_*^\top \mathbf{x}_i)) \mathbf{x}_i \mathbf{x}_i^\top)$.

It is easy to show that $B = (1 + 2\lambda R)/(C_{\min} + \lambda)$ makes $g(\Delta) > 0$. Based on previous argument, since $g(\hat{\Delta}) \leq 0$, we have

$$\begin{aligned} \|\hat{\Delta}\|_2 & \leq \frac{1 + 2\lambda R}{C_{\min} + \lambda} \\ & = \frac{1 + 2\lambda R}{\lambda_{\min} \left(\lambda \mathbf{I} + \frac{1}{n} \sum_{i=1}^n \sigma(\mathbf{w}_*^\top \mathbf{x}_i)(1 - \sigma(\mathbf{w}_*^\top \mathbf{x}_i)) \mathbf{x}_i \mathbf{x}_i^\top \right)} \\ & = (1 + 2\lambda R) \lambda_{\max} \left(\left(\lambda \mathbf{I} + \frac{1}{n} \sum_{i=1}^n D_{ii} \mathbf{x}_i \mathbf{x}_i^\top \right)^{-1} \right). \end{aligned}$$

□

Remark 1: The above bound is derived by analyzing the second-order Taylor expansion of Eq. (1). If we simply use the strongly convex property of Eq. (1), we cannot get the desired bound, because the information of the second-order derivative will not be fully utilized. Consequently, the above bound is sharper than the bound derived by strong convexity.

In logistic regression, the classification of a new example is solely based on its estimated conditional class probability. Therefore, we aim to bound the estimation error of the conditional class probability rather than $(\hat{\mathbf{w}}^\top \mathbf{v} - \mathbf{w}_*^\top \mathbf{v})^2$ as in linear regression. Based on Theorem 1, we can prove the following theorem, which achieves our goal.

Theorem 2. For any fixed sample $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ where y_i follows the conditional distribution as in Eq. (2), and $\|\mathbf{x}_i\|_2 \leq 1$ for $\forall i$. $\hat{\mathbf{w}}$ is the estimated weight vector by logistic regression on S . Then the estimated conditional class probability on a validation set $\{\mathbf{v}_j\}_{j=1}^m$ is upper bounded as

$$\begin{aligned} & \mathbb{E}_{Y|X} \left[\sum_{j=1}^m (\Pr(y|\mathbf{v}_j; \hat{\mathbf{w}}) - \Pr(y|\mathbf{v}_j; \mathbf{w}_*))^2 \right] \\ & \leq C_2 \text{tr} \left(\left(\lambda \mathbf{I} + \frac{1}{n} \mathbf{X} \mathbf{D} \mathbf{X}^\top \right)^{-1} \mathbf{V} \mathbf{\Sigma} \mathbf{V}^\top \right), \end{aligned}$$

where $C_2 = (1 + \lambda R)^2 (\lambda + 1)^2 / \lambda^2$, $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]$ and $\mathbf{V} = [\mathbf{v}_1, \dots, \mathbf{v}_m]$, $\mathbf{\Sigma}$ is a diagonal matrix with diagonal elements defined as follows

$$\Sigma_{jj} = (1 - \sigma(\tilde{\mathbf{w}}^\top \mathbf{v}_j)) \sigma(\tilde{\mathbf{w}}^\top \mathbf{v}_j), \quad (4)$$

with $\tilde{\mathbf{w}} = \mathbf{w}_* + \alpha(\hat{\mathbf{w}} - \mathbf{w}_*)$ for some $\alpha \in [0, 1]$.

Proof. (Sketch of proof): Consider the second-order Taylor expansion of $\sigma(\hat{\mathbf{w}}^\top \mathbf{v}_j)$, we have the following inequality,

$$\sigma(\hat{\mathbf{w}}^\top \mathbf{v}_j) = \sigma(\mathbf{w}_*^\top \mathbf{v}_j) + \sigma(\tilde{\mathbf{w}}^\top \mathbf{v}_j) (1 - \sigma(\tilde{\mathbf{w}}^\top \mathbf{v}_j)) \mathbf{v}_j \cdot \hat{\Delta},$$

where $\tilde{\mathbf{w}} = \mathbf{w}_* + \alpha(\hat{\mathbf{w}} - \mathbf{w}_*) = \mathbf{w}_* + \alpha\hat{\Delta}$ for some $\alpha \in [0, 1]$.

Then we have

$$\begin{aligned} & \mathbb{E}_{Y|X} \left[\sum_{j=1}^m (\Pr(y|\mathbf{v}_j, \hat{\mathbf{w}}) - \Pr(y|\mathbf{v}_j, \mathbf{w}_*))^2 \right] \\ &= \sum_{j=1}^m (\sigma(\hat{\mathbf{w}}^\top \mathbf{v}_j) - \sigma(\mathbf{w}_*^\top \mathbf{v}_j))^2 \\ &= \sum_{j=1}^m (\sigma(\tilde{\mathbf{w}}^\top \mathbf{v}_j) (1 - \sigma(\tilde{\mathbf{w}}^\top \mathbf{v}_j)) \mathbf{v}_j \cdot \hat{\Delta})^2 \\ &= \sum_{j=1}^m (\Sigma_{jj} \mathbf{v}_j \cdot \hat{\Delta})^2, \end{aligned}$$

which can be further bounded by Theorem 1. \square

Remark 2: All the above theoretical results hold under the conditional expectation with respect to the conditional class distribution $\Pr(Y|X)$, given any fixed design matrix \mathbf{X} . They do not require either $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ or $\{\mathbf{x}_i\}_{i=1}^n$ to be i.i.d., which is the common assumption in passive learning. In addition, the derived bounds do not depend on the class labels of the training sample explicitly.

It can be observed from Theorem 2 that, the expected estimation error of the conditional class probability $P(Y|X)$ on a validation set is upper bounded by a term which can be approximately computed based on the training set together with the validation set without their labels. Therefore, they can be used to guide the design of active learning algorithms, because the examples in the pool are not only dependent (starting from the second round of label query) in active learning, but also unlabeled. It also explains why we need to derive such a kind of bounds to design active learning algorithms rather than using existing PAC-style bounds for model-free learning [8]. In a nutshell, we can minimize this bound by choosing a subsample of the training set. We will discuss this in details in the next section.

3 ACTIVE LEARNING BASED ON ERROR BOUND MINIMIZATION

Before presenting the new active learning method, let us recall the basic setting of batch-mode active learning as follows. Given a training data matrix, i.e., $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n] \in \mathbb{R}^{d \times n}$, and an initial labeled set \mathcal{L} , together with a set of unlabeled examples, i.e., \mathcal{U} . Batch-mode active learning operates in T iterations. In each iteration, the

learner will choose b examples (denoted by \mathcal{B}) from the unlabeled set \mathcal{U} to label, and add these labeled examples into the existing labeled set \mathcal{L} (also remove \mathcal{B} from the unlabeled set \mathcal{U}). The goal of batch-mode active learning is to find bT examples in total, which are the most informative examples, namely selected subsample set, to query their labels.

3.1 THE CRITERION

The proposed active learning method is motivated by Theorem 2. In Theorem 2, we can see that the estimation error of the conditional class probability is upper bounded by $\text{tr}(\Sigma^{\frac{1}{2}} \mathbf{V}^\top (\lambda \mathbf{I} + 1/n \mathbf{X} \mathbf{D} \mathbf{X}^\top)^{-1} \mathbf{V}^\top \Sigma^{\frac{1}{2}})$, where \mathbf{D} and Σ are depending on \mathbf{w}_* and $\tilde{\mathbf{w}}$. Since \mathbf{w}_* and $\tilde{\mathbf{w}}$ are unknown, we cannot calculate \mathbf{D} and Σ exactly. Instead, we use the current $\hat{\mathbf{w}}$ to approximate \mathbf{w}_* and $\tilde{\mathbf{w}}$. Based on the approximate \mathbf{D} and Σ , we can choose b examples from \mathcal{U} which minimizes the upper bound. Then we will use these newly labeled b examples together with existing labeled examples to update the classifier. After that, we may get better approximations to \mathbf{D} and Σ . This process is repeated until the label budget is used out.

More specifically, in the t -th iteration, we have labeled set \mathcal{L} and unlabeled set \mathcal{U} . We also have the classifier $\hat{\mathbf{w}}_t$, based on which we can get approximations of \mathbf{D} and Σ . Then we are going to choose the next b examples by minimizing the following criterion,

$$\arg \min_{\mathcal{B} \subset \mathcal{U}} \text{tr} \left(\Sigma^{\frac{1}{2}} \mathbf{V}^\top (\lambda \mathbf{I} + \mathbf{X}_{\mathcal{B}} \mathbf{D}_{\mathcal{B}\mathcal{B}} \mathbf{X}_{\mathcal{B}}^\top)^{-1} \mathbf{V} \Sigma^{\frac{1}{2}} \right),$$

where we absorb $1/n$ into λ . By introducing $\tilde{\mathbf{v}}_j = \sqrt{\Sigma_{jj}} \mathbf{v}_j$ and $\tilde{\mathbf{x}}_i = \sqrt{D_{ii}} \mathbf{x}_i$, the above optimization problem can be simplified as

$$\arg \min_{\mathcal{B} \subset \mathcal{U}} \text{tr} \left(\tilde{\mathbf{V}}^\top (\lambda \mathbf{I} + \tilde{\mathbf{X}}_{\mathcal{B}} \tilde{\mathbf{X}}_{\mathcal{B}}^\top)^{-1} \tilde{\mathbf{V}} \right). \quad (5)$$

It is a combinatorial optimization problem. Similar problems have been encountered in previous work [29]. One way to solve it is applying the sequential minimization algorithm derived in [29] b times, to get a batch \mathcal{B} . However, this sacrifices the advantage of batch-mode active learning, because it neglects the information overlap among examples. Another way is formulating it as a semi-definite programming [9], which is computationally very expensive. Here, we do some relaxation and use the projected gradient descent to solve it, following the idea adopted in [14].

3.2 OPTIMIZATION

We introduce a selection matrix $\mathbf{S} \in \mathbb{R}^{|\mathcal{U}| \times b}$, which is defined as

$$S_{ij} = \begin{cases} 1, & \text{if the } i\text{-th example in } \mathcal{U} \text{ is selected} \\ & \text{as the } j\text{-point in } \mathcal{B} \\ 0, & \text{otherwise.} \end{cases}$$

It is easy to check that each column of \mathbf{S} has one and only one 1, and each row has at most one 1. We denote the constraint set for \mathbf{S} by $\mathcal{S}_1 = \{\mathbf{S} | \mathbf{S} \in \{0, 1\}^{|\mathcal{U}| \times b}, \mathbf{S}^\top \mathbf{S} = \mathbf{I}\}$.

With \mathbf{S} , we have $\tilde{\mathbf{X}}_B = \tilde{\mathbf{X}}\mathbf{S}$. Then Eq. (5) can be simplified as

$$\arg \min_{\mathbf{S} \in \mathcal{S}_1} \text{tr} \left(\tilde{\mathbf{V}}^\top (\lambda \mathbf{I} + \tilde{\mathbf{X}}\mathbf{S}\mathbf{S}^\top \tilde{\mathbf{X}}^\top)^{-1} \tilde{\mathbf{V}} \right).$$

The above optimization problem is almost continuous, except the constraint set \mathcal{S}_1 . In order to apply continuous optimization algorithms, we relax it into the following continuous domain, i.e., $\mathcal{S}_2 = \{\mathbf{S} | \mathbf{S} \geq 0, \mathbf{S}^\top \mathbf{S} = \mathbf{I}\}$.

Since the projection onto $\{\mathbf{S} : \mathbf{S}^\top \mathbf{S} = \mathbf{I}\}$ is computationally expensive, we would like to design an algorithm in which the constraint $\mathbf{S}^\top \mathbf{S} = \mathbf{I}$ is automatically satisfied after each gradient descent. To cope with $\mathbf{S}^\top \mathbf{S} = \mathbf{I}$, we introduce a Lagrange multiplier $\mathbf{\Lambda} \in \mathbb{R}^{b \times b}$, and write down the Lagrangian function as

$$L(\mathbf{S}) = \text{tr} \left(\tilde{\mathbf{V}}^\top (\lambda \mathbf{I} + \tilde{\mathbf{X}}\mathbf{S}\mathbf{S}^\top \tilde{\mathbf{X}}^\top)^{-1} \tilde{\mathbf{V}} \right) + \text{tr} (\mathbf{\Lambda} (\mathbf{S}^\top \mathbf{S} - \mathbf{I})).$$

The derivative of $L(\mathbf{S})$ with respect to \mathbf{S} is

$$\frac{\partial L}{\partial \mathbf{S}} = -2\tilde{\mathbf{X}}^\top \mathbf{B} \tilde{\mathbf{X}} \mathbf{S} + 2\mathbf{S} \mathbf{\Lambda}, \quad (6)$$

where $\mathbf{B} = \mathbf{A}^{-1}(\tilde{\mathbf{V}}\tilde{\mathbf{V}}^\top)\mathbf{A}^{-1}$ and $\mathbf{A} = \lambda \mathbf{I} + \tilde{\mathbf{X}}\mathbf{S}\mathbf{S}^\top \tilde{\mathbf{X}}^\top$. Using the fact that $\mathbf{S}^\top \mathbf{S} = \mathbf{I}$ yields $\mathbf{\Lambda} = \mathbf{S}^\top \tilde{\mathbf{X}}^\top \mathbf{B} \tilde{\mathbf{X}} \mathbf{S}$. Substituting the Lagrange multiplier $\mathbf{\Lambda}$ back into Eq. (6), we obtain the derivative depending solely on \mathbf{S} . Then following [14], we can use projected gradient descent to find a local optimal solution for Eq. (6), where the projection is only onto $\{\mathbf{S} : \mathbf{S} \geq 0\}$. After the local optimal \mathbf{S}^* is obtained, we can discretize it to obtain the desired solution. The analysis of the gap between the local optima and the global optima is challenging and perhaps an open problem. It may be helpful to realize that \mathcal{S}_2 is a matching polytope [23] for such kind of analysis.

In summary, we present the whole algorithm for active learning based on error bound minimization in Algorithm 1. Since our algorithm is designed from logistic regression, we call it **Logistic Bound**. In the special case that $b = 1$, i.e., fully sequential active learning, we do not need to use projected gradient descent in each iteration. In that case, we can find the best single example by sorting.

We emphasize that α in Theorem 2 is some parameter within $[0, 1]$. This parameter comes from the mean value theorem in the derivation. It is not a parameter of our algorithm, because we use $\hat{\mathbf{w}}$ to approximate $\tilde{\mathbf{w}}$ in Algorithm 1. So we do not need to tune α at all.

Algorithm 1 Batch-Mode Active Learning Based on Error Bound Minimization (Logistic Bound)

Input: \mathbf{X} , \mathbf{V} , number of iterations T , batch size b , regularization parameter λ , initial labeled set \mathcal{L} and unlabeled set \mathcal{U} ;

for $t = 1 \rightarrow T$ **do**

 Compute $\hat{\mathbf{w}}_t$ based on \mathcal{L} ;

 Compute \mathbf{D} and $\mathbf{\Sigma}$ based on Eqs. (3) and (4);

 Compute $\mathcal{B} \subset \mathcal{U}$ based on Eq. (5);

 Update $\mathcal{L} = \mathcal{L} \cup \mathcal{B}$ and $\mathcal{U} = \mathcal{U} \setminus \mathcal{B}$;

end for

3.3 TIME COMPLEXITY

In this subsection, we analyze the time complexity of the proposed active learning algorithm. The computation of Eq. (6) involves \mathbf{A}^{-1} , which is the inverse of a $d \times d$ matrix. However, we do not need to compute it directly. Since $\mathbf{A}^{-1} = (\lambda \mathbf{I} + \mathbf{X}\mathbf{S}\mathbf{S}^\top \mathbf{X}^\top)^{-1}$, by applying the Woodbury matrix identity, we have $\mathbf{A}^{-1} = 1/\lambda \mathbf{I} - 1/\lambda \mathbf{X}\mathbf{S}(\lambda \mathbf{I} + \mathbf{S}^\top \mathbf{X}^\top \mathbf{X}\mathbf{S})^{-1} \mathbf{S}^\top \mathbf{X}^\top$. Thus we only need to calculate the inverse of $(\lambda \mathbf{I} + \mathbf{S}^\top \mathbf{X}^\top \mathbf{X}\mathbf{S})$, whose size is $b \times b$, where b is the batch size. So the time complexity of computing the gradient in Eq. (6) can be reduced to $O(ndb + db^2 + b^3)$, which is dominated by $O(ndb)$ because b is often set to 5 to 50. The total complexity of the projected gradient descent is $O(ndbt)$, where t is the iteration number. The time complexity is clearly linear to the sample size n , and the dimension of the input space d .

4 RELATED WORK

In this section, we show the connections of the proposed approach with some existing active learning methods.

One thread of related work is experimental design [2]. For instance, Yu et al. [29] proposed transductive experimental design (TED), whose intent is to select the examples to learn a least squares regression function which has minimum prediction variance on the validation data. As we mentioned before, it is a *non-adaptive* active learning method, because the label information of the selected examples cannot be utilized to select subsequent unlabeled examples. Intuitively, taking into account the labels of queried examples is beneficial for subsequent label query. Our method is able to utilize the labeled examples obtained up to now to choose the next batch of examples through \mathbf{D} and $\mathbf{\Sigma}$. Recall that in our method, each example in the pool is weighted by $D_{ii} = \sigma(\hat{\mathbf{w}}^\top \mathbf{x}_i)(1 - \sigma(\hat{\mathbf{w}}^\top \mathbf{x}_i))$. Apparently, the more uncertain an example is, the bigger its weight will be (because D_{ii} is maximized when $\sigma(\hat{\mathbf{w}}^\top \mathbf{x}_i) = 1/2$). In the special case, if $\sigma(\hat{\mathbf{w}}^\top \mathbf{x}_i) = 1/2$ for every example in the pool, then all the examples are equally weighted, and our method will degrade to TED. By applying the derivation technique to ridge regression, we

can obtain a very similar result to [29]. However, using the derivation technique from experimental design, we cannot get the results in this paper. So the derivation technique used in this paper is more general.

The second line of related work is active learning based on logistic regression [30, 26]. Based on the asymptotic analysis, Zhang and Oles [30] derived the inverse of the Fisher information matrix of the maximum likelihood estimation (MLE) of logistic regression, which measures the variance of model. Thus they proposed an active learning criterion by minimizing the variance. Our criterion derived in Theorem 2 is from finite sample analysis, which is non-asymptotic and different from the criterion derived in [30]. Since finite sample error bounds characterize the behavior of a classifier provided with a finite training sample, they provide more accurate guidance on algorithm design than asymptotic analysis. Following this seminal work, several incremental studies were presented, which solve the same criterion using different sophisticated optimization algorithms. For example, Hoi et al. [20] proposed to reformulate it as a submodular function maximization problem. On the other hand, Guo and Schuurmans [17] proposed a batch-mode active learning algorithm based on logistic regression, by maximizing the likelihood on the labeled training sample and minimizing the entropy on the selected unlabeled training sample. The main innovation lies in the optimization part rather than the theoretical results.

The last but not least related work is the family of expected error minimization-based approaches. Recently, Gu et al. [14] proposed an active learning method based on minimizing the out-of-sample error bound for Laplacian regularized least squares (LapRLS) [6], a semi-supervised version of least squares regression. It sheds light on designing active learning algorithms via deriving certain kind of error bounds, which do not rely on the i.i.d assumption of the training sample nor the class labels. The method is non-adaptive¹, raising a question that whether we can design an adaptive active learning algorithm along this line. Our result in this paper is in the affirmative. However, the linear regression model as well as the derivation technique used there are not capable to achieve this goal. So we study logistic regression with a new analyzing technique instead. Note also that in the supervised case, the methods proposed in [14] and [29] are identical in terms of the criterion.

¹It is nonadaptive, in the sense that when the model ($\hat{\mathbf{w}}$) is updated using the newly labeled examples, the algorithm is not able to use the information from the updated model ($\hat{\mathbf{w}}$) to choose next batch of examples to label. In fact, the active learning algorithm in [14] does not use any information from $\hat{\mathbf{w}}$ in the process of active learning, because its criterion does not depend on $\hat{\mathbf{w}}$.

5 EXPERIMENTS

In this section, we evaluate the proposed method on both UCI datasets [3] (wdbc, wpbc, sonar, heart, australian, diabetes, splice) and two text datasets (Text1 and Text2) generated from the famous 20-newsgroups data set². For the text datasets, the original number of features (words) is 8,014. We apply principal component analysis (PCA) to reduce the input dimensionality by projecting the data onto its leading principal components, where the number of principal component is determined such that it accounts for 95% of its total variance. For each example, we normalize it into a vector with unit ℓ_2 -norm.

5.1 EXPERIMENTAL SETUP AND BASELINES

In order to randomize the experiments, in each run of experiments, we use 50% data as the training examples. The remaining 50% data is used as test set. We use the training set for active learning and evaluate the prediction performance on the fixed test set. This random split was repeated 20 times, thus we can do statistical significance test. We study a difficult case of active learning, where we start with one randomly selected example per class. All the algorithms start with the same initial labeled set, unlabeled set and test set.

To demonstrate the effectiveness of our proposed method, we compare it with existing state-of-the-art algorithms, including one fully sequential active learning approach, one non-adaptive active learning algorithm, and three batch-mode active learning methods. We summarize these methods as follows:

Random Sampling (Random): It is the simplest baseline, which uniformly selects examples from the candidate set as training data.

Query the informative and representative examples (QUIRE) [22]: it is a fully sequential active learning algorithm.

Transductive experiment design (TED) [29]: it is a non-adaptive active learning method. Note that it selects all the examples to label at one shot.

SVM batch-mode active learning (SVM BMAL) [21]: in our empirical study, we found that it consistently outperforms SVM active learning [27], so we only demonstrate its results while omit the results of SVM active learning. We use linear kernel for SVM. In fact, SVM active learning can be seen as a special case of SVM BMAL, where the batch size is equal to 1.

Discriminative batch-mode active learning (Disc) [17]: it is a batch-mode active learning algorithm based on logistic regression.

²<http://people.csail.mit.edu/jrennie/20NewsGroups/>

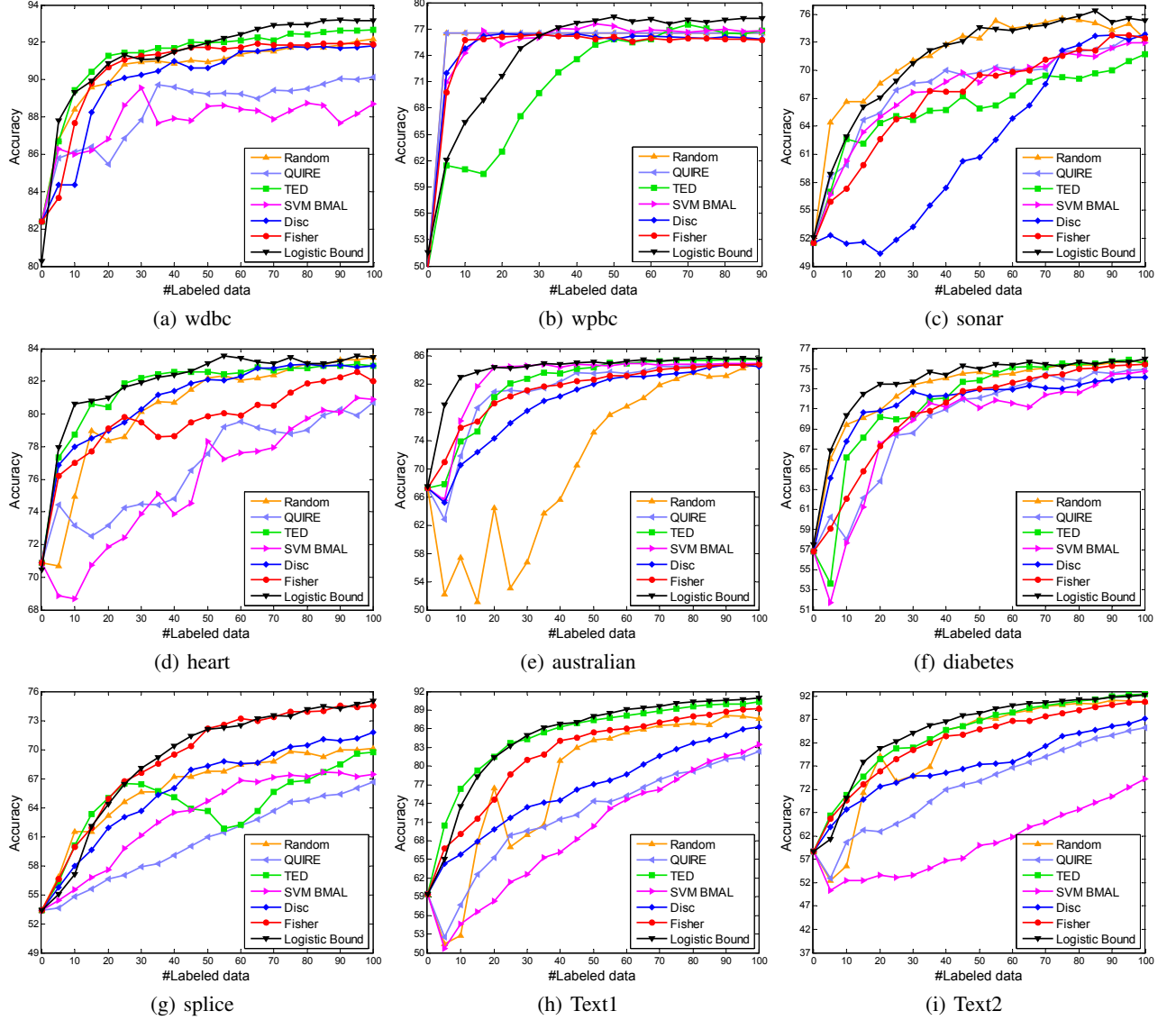


Figure 1: Comparison of active learning methods on both UCI and text datasets with batch size $b = 5$, and $T = 20$ iterations. The x-axis is number of labeled examples, and the y-axis is the classification accuracy (%).

Table 1: Win/tie/loss counts for the proposed method versus the other methods during the whole active learning process, based on paired t-test at 95% significance level. The first column is the dataset name (#examples/#features).

DATASETS	VS RANDOM	VS QUIRE	VS TED	VS SVM_BMAL	VS DISC	VS FISHER
WDBC(569/30)	14/6/0	18/2/0	8/12/0	18/2/0	17/3/0	10/10/0
WPBC(198/33)	9/6/5	9/6/5	13/7/0	4/12/4	10/6/4	11/6/3
SONAR(208/60)	2/16/2	13/7/0	18/2/0	14/6/0	20/0/0	19/1/0
HEART(270/13)	9/11/0	19/1/0	6/13/1	20/0/0	5/15/0	14/6/0
AUSTRALIAN(690/14)	14/6/0	20/0/0	10/10/0	10/10/0	20/0/0	20/0/0
DIABETES(768/8)	1/19/0	20/0/0	11/9/0	20/0/0	17/3/0	15/5/0
SPLICE(1000/60)	15/4/1	18/2/0	14/5/1	18/2/0	16/4/0	0/19/1
TEXT1(1980/991)	20/0/0	20/0/0	13/5/2	20/0/0	19/1/0	19/1/0
TEXT2(1990/768)	18/2/0	20/0/0	9/10/1	20/0/0	18/2/0	18/1/1

Fisher information of logistic regression (**Fisher**) [20]: It is also a batch-mode active learning based on logistic regression. However, it is derived from non-asymptotic analysis of logistic regression.

For our method, the validation set \mathbf{V} is set to the same as the pool of unlabeled examples. Recall that our method does not require the labels of the validation set either.

For each dataset, we let the active learning methods incrementally choose $b = 5$ examples to label, and perform $T = 20$ iterations in total (except for wpbc, where we only perform $T = 19$ iterations due to limited examples). We did not compare with [16], because we were not able to acquire a working implementation of this algorithm. According to the experimental results (Table 1) reported in [16], its performance is statistically similar to Disc [17]. We did not use semi-supervised classifiers. Hence the approach proposed in [14] reduces to TED. Most of the implementations are provided by the authors of the corresponding papers.

One issue with most of the active learning methods we investigated is that they are invented based on different classifiers. For example, TED is designed for ridge regression. Disc and Fisher are developed based on logistic regression. We use different classifiers for different active learning approaches, because we found that using the classifier based on which the active learning method is derived can lead to better results than using other classifiers. Furthermore, for each active learning method, its parameter and the parameter of its corresponding classifier are tuned by 5-fold cross validation on the labeled set through searching the grid $\{10^{-3}, 10^{-2}, \dots, 10^3\}$.

5.2 RESULTS AND DISCUSSIONS

The experimental results are shown in Figure 1. In all sub-figures, the x-axis represents the number of labeled examples, while the y-axis is the averaged classification accuracy on the test data over 20 runs.

We compare all the active learning methods during the entire query process. Recall that in Figure 1, there are 20 query points (except for wpbc, which has only 19), with 20 results on each of them. We therefore run a 2-sided paired t-test at each query point, at 95% significance level. The results of t-test can be categorized into three cases: (i) our method outperforms a specific algorithm significantly, denoted by “win”; (ii) our method is significantly worse than a specific algorithm, denoted by “lose”; (iii) otherwise, denoted by “tie”. We summarize the t-test results in terms of the count of “win”, “tie” and “lose” in Table 1.

We observe that the proposed method outperforms the other methods significantly at most cases. SVM_BMAL and QUIRE are often the worst. The reason is probably that their criteria are not related to prediction performance. The

performance of Disc is satisfactory. Yet it performs well on some datasets while not very well on other datasets. The performance of TED and Fisher are comparable. Although TED aims to minimize the variance of prediction, [14] showed that it is actually consistent with minimizing the out-of-sample error of ridge regression. This explains its good performance. However, since TED is a nonadaptive active learning method, it cannot fully utilize the label information during the query process. This limits its performance on many datasets. Fisher minimizes the uncertainty of the model, which does not necessarily lead to small generalization error. However, it happens that the uncertain reduction criterion for logistic regression derived in [30] is a little similar to our criterion. This may interpret its general good performance. The superior performance of our method is attributed to its theoretical foundation, which guarantees that the classifier can achieve small prediction error on the unseen data. Lastly, we found that the performance of random sampling is not bad. As an unbiased label selection procedure, random sampling is at least a consistent algorithm to choose the training sample, as is widely done in passive learning. This is consistent with the result reported in [17].

5.3 STUDY ON THE BATCH SIZE

In previous experiments, we fixed the batch size to 5, which could be biased in comparison. So we will compare our method with those batch-mode active learning algorithms under different settings of batch size here. We vary the batch size using the grid $\{1, 5, 10, 20, 30, 60\}$ and show the results with 60 labeled examples in Figure 2. We only show the results on three datasets (Sonar, Heart and Text1). Similar results can be observed on the other datasets.

It can be seen that under different batch sizes, our method outperforms the other batch-mode active learning algorithms in most cases. This strengthens the superiority of our method over the others. In addition, we also observe some interesting results. For example, for some batch-mode active learning algorithms such as SVM_BMAL and Disc, their performance of using a batch size of more than one example sometimes seems not as good as choosing a single example at each round. This implies that they may not be able to address the information overlap among examples very well. In contrast, our method is able to exploit the interdependence among examples, because our method usually achieves better results with batch-size larger than one.

In addition, we found that our method obtains the best result when the batch size is either not too small ($b = 1$) nor too large ($b = 60$). This is quite reasonable, because when $b = 1$, it is a fully sequential strategy, and we cannot utilize the dependence among examples. On the contrary, if the batch size is too large (such as one-shot active learning in

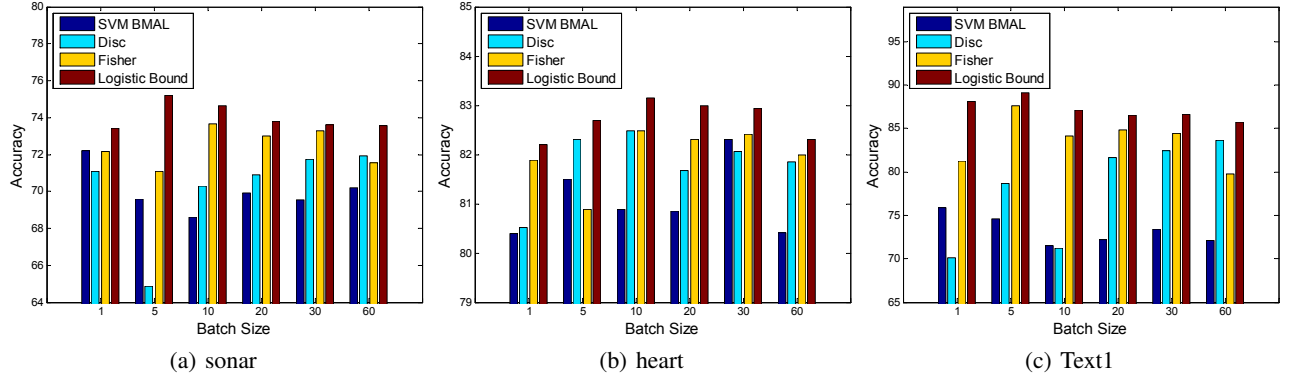


Figure 2: Comparison of batch-mode active learning methods on three datasets with different batch size ranging from $b = 1$ to $b = 60$. The x-axis represents the batch size, and the y-axis is the classification accuracy (%) with 60 labeled examples.

the extreme case), the information contained in the newly labeled examples cannot be immediately exploited through updating the classifier, which may limit its performance. This somehow implies the superiority of batch-mode active learning against both fully sequential and one-shot active learning. It also suggests us to choose a medium size of batch in practice. More rigorous analysis is required in the future work.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we present a novel active learning method based on out-of-sample error bound minimization. We use logistic regression as a running example to derive the algorithm. We would like to emphasize that the derivation technique developed in this paper applies to other generalized linear models, or even more sophisticated graphical models. In our future work, we will study these alternatives. We also plan to conduct comparisons with some other batch-mode active learning methods proposed recently [4, 10, 28]. On the other hand, we aim to develop an algorithm solving Eq. (5) with provable guarantee.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments. This work was supported in part by the U.S. Army Research Laboratory under Cooperative Agreement No. W911NF-09-2-0053 (NS-CTA), the U.S. Army Research Office under Cooperative Agreement No. W911NF-13-1-0193, U.S. National Science Foundation grants CNS-0931975, IIS-1017362, IIS-1320617, IIS-1354329, DTRA, NASA NRA-NNH10ZDA001N, and MIAS, a DHS-IDS Center for Multimodal Information Access and Synthesis at UIUC. The first author was supported by IBM Ph.D. Fellowship (2013-2014).

References

- [1] A. Agarwal. Selective sampling algorithms for cost-sensitive multiclass prediction. In *ICML (3)*, pages 1220–1228, 2013.
- [2] A. D. Anthony Atkinson and R. Tobias. *Optimum Experimental Designs*. Oxford Statistical Science Series. Oxford University Press, May 2007.
- [3] A. Asuncion and D. Newman. UCI machine learning repository, 2007.
- [4] J. Azimi, A. Fern, and X. Fern. Batch bayesian optimization via simulation matching. In *NIPS*, pages 109–117, 2010.
- [5] M.-F. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. In *ICML*, pages 65–72, 2006.
- [6] M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.
- [7] A. Beygelzimer, D. Hsu, J. Langford, and T. Zhang. Agnostic active learning without constraints. In *NIPS*, pages 199–207, 2010.
- [8] O. Bousquet, S. Boucheron, and G. Lugosi. Introduction to statistical learning theory. In *Advanced Lectures on Machine Learning*, pages 169–207, 2003.
- [9] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, Cambridge, 2004.
- [10] Y. Chen and A. Krause. Near-optimal batch mode active learning and adaptive submodular optimization. In *International Conference on Machine Learning (ICML)*, 2013.
- [11] D. A. Cohn, L. E. Atlas, and R. E. Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, 1994.
- [12] S. Dasgupta, D. Hsu, and C. Monteleoni. A general agnostic active learning algorithm. In *NIPS*, 2007.

- [13] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28(2-3):133–168, 1997.
- [14] Q. Gu, T. Zhang, C. H. Q. Ding, and J. Han. Selective labeling via error bound minimization. In *NIPS*, pages 332–340, 2012.
- [15] A. Guillory and J. A. Bilmes. Label selection on graphs. In *NIPS*, pages 691–699, 2009.
- [16] Y. Guo. Active instance sampling via matrix partition. In *NIPS*, pages 802–810, 2010.
- [17] Y. Guo and D. Schuurmans. Discriminative batch mode active learning. In *NIPS*, 2007.
- [18] S. Hanneke. Rates of convergence in active learning. *The Annals of Statistics*, 39(1):333–361, 2011.
- [19] T. Hastie, R. Tibshirani, and J. H. Friedman. *The elements of statistical learning: data mining, inference, and prediction: with 200 full-color illustrations*. New York: Springer-Verlag, 2001.
- [20] S. C. H. Hoi, R. Jin, J. Zhu, and M. R. Lyu. Batch mode active learning and its application to medical image classification. In *ICML*, pages 417–424, 2006.
- [21] S. C. H. Hoi, R. Jin, J. Zhu, and M. R. Lyu. Semi-supervised svm batch mode active learning for image retrieval. In *CVPR*, 2008.
- [22] S.-J. Huang, R. Jin, and Z.-H. Zhou. Active learning by querying informative and representative examples. In *NIPS*, pages 892–900, 2010.
- [23] B. Korte and J. Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer Publishing Company, Incorporated, 4th edition, 2007.
- [24] A. Krause, A. P. Singh, and C. Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9:235–284, 2008.
- [25] P. Ravikumar, M. J. Wainwright, and J. D. Lafferty. High-dimensional ising model selection using ℓ_1 -regularized logistic regression. *The Annals of Statistics*, 38(3):1287C1319, 2010.
- [26] A. I. Schein and L. H. Ungar. Active learning for logistic regression: an evaluation. *Machine Learning*, 68(3):235–265, 2007.
- [27] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. In *ICML*, pages 999–1006, 2000.
- [28] Z. Wang and J. Ye. Querying discriminative and representative samples for batch mode active learning. In *KDD*, pages 158–166, 2013.
- [29] K. Yu, J. Bi, and V. Tresp. Active learning via transductive experimental design. In *ICML*, pages 1081–1088, 2006.
- [30] T. Zhang and F. J. Oles. A probability analysis on the value of unlabeled data for classification problems. In *ICML*, 2000.

Efficient Bayesian Nonparametric Modelling of Structured Point Processes

Tom Gunter*

Chris Lloyd*

Michael A. Osborne

Stephen J. Roberts

Machine Learning Research Group
Department of Engineering Science
University of Oxford
{tgunter, clloyd, mosb, sjrob}@robots.ox.ac.uk

Abstract

This paper presents a Bayesian generative model for dependent Cox point processes, alongside an efficient inference scheme which scales as if the point processes were modelled independently. We can handle missing data naturally, infer latent structure, and cope with large numbers of observed processes. A further novel contribution enables the model to work effectively in higher dimensional spaces. Using this method, we achieve vastly improved predictive performance on both 2D and 1D real data, validating our structured approach.

1 INTRODUCTION

Point processes are effectively used to model a variety of event data, and have also shown a recent popularity within the Machine Learning community as priors over sets. The most fundamental example of such a stochastic model for random sets is the homogenous Poisson process. This is defined via an intensity which describes the expected number of points found in any bounded region of some arbitrary domain. An inhomogenous Poisson process allows the intensity to vary throughout the domain over which the process is defined. As we do not know the functional form of this intensity given only event data, another stochastic process is typically used to model it nonparametrically. This is then termed a doubly-stochastic Poisson process, a type of renewal process also known as a Cox process. In our particular construction, we use transformed Gaussian processes to model the intensity functions of the individual dependent point processes, in such a manner as to enable fully nonparametric Bayesian inference (Adams et al., 2009; Murray et al., 2010). While we only explicitly consider the doubly

stochastic Poisson process, any general renewal process (Rao and Teh, 2011) could be incorporated into the framework we define.

There are many occasions when we have multiple point processes which we expect to be dependent: If the domain is temporal, then an example would be individual clients making trades with a specific financial services provider, or individual customers purchasing items from a specific vendor. If the domain is spatial, we might consider different categories of crime defined over some geographic region. Defining a flexible model for inter-process dependency structure, alongside an efficient inference scheme allows us to learn the underlying intensity functions which drive the typical behaviour. These can then be used to make more accurate predictions, especially during periods of unobservability for an individual process.

In order to maximise the flexibility of our approach, we specifically assume that the individual intensity functions arise via a weighted summation of convolutions of latent functions with a kernel. Intuitively this means that we take a small number of latent functions, individually smooth and scale them, and then add them together to yield an intensity function. This approach allows a wide range of intensities to arise from only a few latent functions.

We will present and validate the following novel contributions:

- The first generative model for dependent Cox process data (Section 2).
- An efficient, parallelised inference scheme, which scales benignly with the number of observed point processes (Section 3).
- A new adaptation of thinning (Lewis, 1979), which we term ‘adaptive thinning’. This introduces multiple uniformisation levels over the space, making the model viable for higher dimensional spaces and larger datasets (Section 4).

* Corresponding authors, in alphabetical order.

2 THE MODEL

We first formally review the Cox process, before describing the innovative nonparametric Bayesian model outlined in Adams et al. (2009) as the Sigmoidal Gaussian Cox Process (SGCP), which allows a full Gaussian process to be used as a prior over an individual intensity function. We then move on to review the convolution process (Álvarez and Lawrence, 2011), a method of modelling dependent functions and the underlying latent processes which govern them. Our novel combination of these constituent elements represents the first model for dependent Cox point processes.

2.1 THE INHOMOGENOUS POISSON PROCESS

For a domain $\mathcal{X} = \mathbb{R}^D$ of arbitrary dimension D , we may define an inhomogenous Poisson process via an intensity function $\lambda(x) : \mathcal{X} \rightarrow \mathbb{R}^+$, and a Lebesgue measure over the domain, dx . The number of events $N(\mathcal{T})$ found over a subregion $\mathcal{T} \subset \mathcal{X}$ will be Poisson distributed with parameter $\lambda_{\mathcal{T}} = \int_{\mathcal{T}} \lambda(x) dx$. Furthermore, we define $N(\mathcal{T}_i)$ to be independent random variables, where \mathcal{T}_i are disjoint subsets of \mathcal{X} (Kingman, 1993).

If we bound the region to be considered, and assume there are K observed events, labelled as $\{x_k\}_{k=1}^K$, then the inhomogenous Poisson process likelihood function may be written as

$$p(\{x_k\}_{k=1}^K | \lambda(x)) = \exp \left\{ - \int_{\mathcal{T}} dx \lambda(x) \right\} \prod_{k=1}^K \lambda(x_k). \quad (1)$$

2.2 THE SIGMOIDAL GAUSSIAN COX PROCESS

In order to model the intensity nonparametrically, we place a Gaussian Process (Rasmussen and Williams, 2006) prior over a random scalar function $g(x) : \mathcal{X} \rightarrow \mathbb{R}$. This means that the prior over any finite set of function values $\{g(x_n)\}_{n=1}^N$ is a multivariate Gaussian distribution, defined by a positive definite covariance function $C(.,.) : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ and a mean function $m(.) : \mathcal{X} \rightarrow \mathbb{R}$. The mean and covariance function are parameterised by a set of hyperparameters, which we denote γ .

In the SGCP, a Gaussian Process is transformed into a prior over the intensity function by passing it through a sigmoid function and scaling it against a maximum intensity λ^* : $\lambda(x) = \lambda^* \sigma(g(x))$, where $\sigma(.)$ is the logistic function. This forms the basis of a generative prior, whereby exact Poisson data can be generated from $\lambda(x)$ via thinning (Lewis, 1979), which involves

adding M events, such that the joint point process over the $M + K$ events is homogenous with fixed rate λ^* .

As we are using an infinite dimensional proxy for $\lambda(x)$, the integral in Equation 1 is intractable. Furthermore, using Bayes' theorem with this likelihood yields a posterior with intractable integrals in both the numerator and denominator. These challenges are overcome by making use of the generative prior, and augmenting the variable set to include the number of thinned points, M , and their locations, $\{\tilde{x}_m\}_{m=1}^M$. This then means that the value of the intensity function need only be inferred at the $M + K$ point locations, $\mathbf{g}_{\mathbf{M}+\mathbf{K}} = \{g(x_k)\}_{k=1}^K \cup \{g(\tilde{x}_m)\}_{m=1}^M$. Noting that $\sigma(-z) = 1 - \sigma(z)$, the joint likelihood over the data, function values and latent variables is

$$\begin{aligned} p(\{x_k\}_{k=1}^K, M, \{\tilde{x}_m\}_{m=1}^M, \mathbf{g}_{\mathbf{M}+\mathbf{K}} | \lambda^*, \mathcal{T}, \theta) = \\ (\lambda^*)^{M+K} \exp \{-\lambda^* \mu(\mathcal{T})\} \\ \times \prod_{k=1}^K \sigma(g(x_k)) \prod_{m=1}^M \sigma(-g(\tilde{x}_m)) \\ \times \mathcal{GP}(\mathbf{g}_{\mathbf{M}+\mathbf{K}} | \{x_k\}_{k=1}^K, \{\tilde{x}_m\}_{m=1}^M, \gamma), \end{aligned} \quad (2)$$

where we have defined $\mu(\mathcal{T}) = \int_{\mathcal{T}} dx$.

Notably, this likelihood equation does not involve any intractable integrals. This means that inference is now possible in this model, albeit subject to the cost of an augmented variable set.

2.3 THE CONVOLUTION PROCESS

The convolution process framework is an elegant way of constructing dependent output processes. Instead of assuming the typical instantaneous (Teh et al., 2005) mixing of a set of independent processes to construct correlated output processes, we generalise to allow a blurring of the latent functions achieved via convolution with a kernel, $G(x, z)$, prior to mixing. z is typically defined on the same domain as x . If we place a Gaussian process prior over the latent function, the output function turns out to also be a Gaussian process (Álvarez and Lawrence, 2011). Specifically, given D dependant intensity functions $g_d(x)$ and Q latent processes $u_q(x)$, (where typically $Q < D$), the stochastic component of the d th intensity is

$$g_d(x) = \sum_{q=1}^Q \int_{\mathcal{T}} G_d(x, z) u_q(z) dz. \quad (3)$$

Given full knowledge of the latent functions, the $g_d(x)$ are independent and deterministic. The $G_d(x, z)$ encode the observed process specific characteristics, and the $u_q(z)$ can be thought of as encoding the latent driving forces.

The convolution process has strong links with the Bayesian kernel method, as described in (Pillai et al., 2007). This allows a function $f(x)$ on \mathcal{X} to arise as

$$f(x) = \int_{\mathcal{X}} K(x, z) U(dz), \quad (4)$$

where $U(dz) \in \mathcal{M}(\mathcal{X})$ is a signed measure on \mathcal{X} . The integral operator $\mathcal{L}_K : U(dz) \rightarrow f(x)$ maps the space of signed measures $\mathcal{M}(\mathcal{X})$ into \mathcal{H}_K , a reproducing kernel Hilbert space (RKHS) defined by the kernel, $K(x, z)$. This mapping is dense in \mathcal{H}_K . If we place a Gaussian process prior on the random signed measure $U(dz)$, rather than directly over $f(x)$, then any draw of $f(x)$ will provably lie in \mathcal{H}_K . As \mathcal{H}_K is equivalent to the span of functions expressible as kernel integrals, this approach allows us to properly construct distributions over specific parts of function space. Using prior domain knowledge to restrict inference to plausible areas of function space is valuable, particularly for point process intensities where the likelihood linking function to data is weak and non-trivial, and we wish to only consider smooth intensities while using a sampling based inference scheme.

The convolution process is also known as a latent force model (Alvarez et al., 2009). In this guise, it is used to infer the solution of a differential equation when there is uncertainty in the forcing function. The convolution kernel is the Green's function of a particular differential equation, and the Gaussian process prior is placed on the driving function. This representation lets us consider the latent functions as driving forces, which are viewed through the intensity function specific convolution kernel. The convolution kernel can, for example, be used to model differing speeds of information propagation from the latent factors to each of the observed processes.

It is worth noting that any general Lévy process prior can be used over the latent functions, however in this particular case we use a pure Gaussian process primarily for reasons of tractability.

2.4 SPARSE LATENT FUNCTIONS

To make the model tractable, we make use of the property that the intensities are independent conditioned on the latent functions. This is made clear from the perspective of a generative model with only one latent function: we first draw a sample of the object $u(z)$, before solving the integral in equation 3, where uncertainty about $u(z)$ is propagated through the convolution. Now instead of maintaining the full, infinite dimensional object $u(z)$, let us condition on a finite dimensional draw of $u(z)$, $u(Z) = [u(z_1), \dots, u(z_J)]^T$ where $Z = \{z_j\}_{j=1}^J$. We can then sample from $p(u(z) | u(Z))$, as this is a conditional Gaussian distribution,

and use this function to solve the convolution integral. With multiple latent functions we can approximate each $u_q(z)$ by $\mathbb{E}[u_q(z) | u_q(Z)]$, replacing Equation 3 with

$$g_d(x) \approx \sum_{q=1}^Q \int_{\mathcal{T}} G_d(x, z) \mathbb{E}[u_q(z) | u_q(Z)] dz. \quad (5)$$

This is reasonable as long as each $u_q(z)$ is smooth, in the sense that it is well approximated given the covariance function and the finite dimensional sample $u_q(Z)$. In Section 3, we use the approximation in Equation 5 along with the conditional independence assumption to build a tractable inference scheme.

2.5 CONSTRUCTING THE MODEL

Let the Q latent functions $u_q(z)$ be modelled as Gaussian processes with Gaussian covariance functions such that

$$u_q | \phi_q \sim \mathcal{GP}(0, K_q(z, z')), \quad (6)$$

where $K_q(z, z')$ is simply the Gaussian kernel

$$K_q(z, z') = \mathcal{N}(z; z', \phi_q). \quad (7)$$

We use a scaled Gaussian convolution kernel

$$G_d(x, z) = \kappa_d \mathcal{N}(x; z, \theta_d). \quad (8)$$

This restricts $g_d(x)$ to be at least as smooth as the random draws from $u_q(z)$. The covariance linking $u_q(z)$ to $g_d(x)$ is

$$\begin{aligned} K_{f_d, u_q}(x, z) &= \int_{\mathcal{X}} G_d(x, z) K_q(z, z') dz \\ &= \kappa_d \mathcal{N}(x; z, \theta_d + \phi_q), \end{aligned} \quad (9)$$

and the overall covariance between output functions is

$$\begin{aligned} K_{f_d, f_{d'}}(x, x') &= \sum_{q=1}^Q \int_{\mathcal{X}} G_d(x, z) \\ &\quad \int_{\mathcal{X}} G_{d'}(x', z') K_q(z, z') dz' dz \\ &= \sum_{q=1}^Q \kappa_d \kappa_{d'} \mathcal{N}(x; z, \theta_d + \theta'_{d'} + \phi_q). \end{aligned} \quad (10)$$

We could use this joint covariance function to construct one large joint Gaussian process over all the intensity functions. In doing this, however, the $u_q(z)$ have been implicitly integrated out, and the resulting inference problem will scale computationally as $\mathcal{O}(D^3 N^3)$ with storage requirements of $\mathcal{O}(D^2 N^2)$, where $N = M + K$ is the joint number of events. This

is intractable for any real problem, where we would hope to leverage many dependent point processes to learn a few latent factors with minimal uncertainty.

Let us now define some additional notation: \mathbf{K} denotes a covariance matrix obtained by evaluating the appropriate covariance function at all eligible pairs of data points. Subscripts determine which covariance is used and hence which inputs are valid, e.g. \mathbf{K}_{g_d, u_q} denotes the cross covariance between the d th output and q th input function. $\mathbf{K}_{g_d, u}$ means stack the Q \mathbf{K}_{g_d, d_q} matrices vertically, $\mathbf{K}_{u, u}$ is a block diagonal matrix where each block corresponds to \mathbf{K}_{u_q, u_q} , and \mathbf{u} is the result of stacking the draws from the finite dimensional Gaussians $p(u_q(Z))$ vertically.

We also define: $\phi = \{\phi_q\}_{q=1}^Q$, $\kappa = \{\kappa_d\}_{d=1}^D$, $\theta = \{\theta_d\}_{d=1}^D$, $X_d = \{x_{d,k}\}_{k=1}^{K_d} \cup \{\tilde{x}_{d,m}\}_{m=1}^{M_d}$. If we wish to allow the latent functions to be sampled at different points, then we define a separate Z_q for each $u_q(z)$: $Z_q = \{z_{q,j}\}_{j=1}^J$. The set of inputs over all latent functions is then $Z = \{Z_q\}_{q=1}^Q$, and similarly for the intensities: $X = \{X_d\}_{d=1}^D$.

Notation in place, we determine that given the approximation in Equation 5, the conditional likelihood for $g_d(x)$ is

$$p(g_d | u, Z, X_d, \kappa_d, \theta_d, \phi) = \mathcal{N}(\mathbf{K}_{g_d, u} \mathbf{K}_{u, u}^{-1} \mathbf{u}, \mathbf{K}_{g_d, g_d} - \mathbf{K}_{g_d, u} \mathbf{K}_{u, u}^{-1} \mathbf{K}_{g_d, u}^T). \quad (11)$$

Still conditioning on the latent functions, the joint likelihood over all D intensity functions is then simply

$$p(g_1, \dots, g_D | u, Z, X, \kappa, \theta, \phi) = \prod_{d=1}^D p(g_d | u, Z, X_d, \kappa_d, \theta_d, \phi). \quad (12)$$

Bayes' rule for Gaussians gives us the posterior over the $u_q(Z)$ as

$$p(u_1, \dots, u_Q | g_1, \dots, g_D, Z, X, \kappa, \phi, \theta) = \mathcal{N}(u_1, \dots, u_Q; \mu_p, \Sigma_p). \quad (13)$$

Where the mean and covariance are

$$\Sigma_p = [\mathbf{K}_{u, u}^{-1} + (\mathbf{K}_{g, u} \mathbf{K}_{u, u}^{-1})^T \mathbf{D}^{-1} (\mathbf{K}_{f, u} \mathbf{K}_{u, u}^{-1})]^{-1} \\ \mu_p = \Sigma_p (\mathbf{K}_{g, u} \mathbf{K}_{u, u}^{-1})^T \mathbf{D}^{-1} \mathbf{g} \quad (14)$$

and where $\mathbf{D} = \mathbf{K}_{g, g} - \mathbf{K}_{g, u} \mathbf{K}_{u, u}^{-1} \mathbf{K}_{g, u}^T$.

The exact form of \mathbf{D} depends on the degree to which we are willing make independence assumptions in order to approximate the Gaussian processes used to model the functions. Naturally the higher the degree of approximation, the more scalable the resulting inference scheme.

Under full dependence, a single event is linked to both inter and intra-process data. We will be assuming that the dependency structure across the $g_d(x)$ is entirely contained by the latent processes, $u_q(z)$. Intuitively, this means that we maintain the full Gaussian process structure for each individual intensity function, while summarising the latent functions via a set of inducing inputs $Z = \{z_j\}_{j=1}^J$. This approximation scheme results in a functional form which is similar to what Quiñero Candela and Rasmussen (2005) call the Partial Independence (PITC) scheme. Importantly it allows inference to scale computationally as $\mathcal{O}(DN^3)$, with storage requirements of $\mathcal{O}(DN^2)$ even in the worst case scenario of $J = N$. Further approximations may be made, and these are especially useful if the number of events per process is large, however for our purposes they are not necessary. For more information on approximation methods for Gaussian processes see Quiñero Candela and Rasmussen (2005) and Snelson and Ghahramani (2005).

Under the PITC low rank covariance, the resulting form for \mathbf{D} is: $[\mathbf{K}_{g, g} - \mathbf{K}_{g, u} \mathbf{K}_{u, u}^{-1} \mathbf{K}_{g, u}^T] \circ \mathbf{M}$, where $\mathbf{M} = \mathbf{I}_N \otimes \mathbf{1}_N$ and $\mathbf{1}_N$ is a $N \times N$ matrix of ones. This may be more familiar as `blkdiag`[\mathbf{D}].

3 INFERENCE

For each of the D point processes we need to learn $|X_d|$, X_d , κ_d , θ_d , λ_d^* and $g_d(x)$. For each of the Q latent functions $u_q(Z)$ and ϕ_q must be inferred. Z_q are fixed to an evenly spaced grid which is identical across the latent processes. To find posteriors over all these variables, we choose a Markov Chain Monte Carlo (MCMC) algorithm, as detailed below.

Using the PITC approximation scheme, the likelihood over the point processes factorises conditioned on the latent functions. This means that given D compute units the updates associated with each point process may be made in parallel. This is important as the inference algorithm is computationally bottlenecked by operations associated with learning the locations of the thinning points, X .

We now give a recap of the inference scheme from the SGCP for a single point process, while listing our minor modifications. Updates for the latent functions are then given, conditioning on the D intensity functions.

3.1 LEARNING THE INTENSITY FUNCTION

Recalling Equation 2, three kinds of Markov transitions are used to draw from this joint distribution: 1) Sampling the number of thinned points, M . 2) Sampling the locations of the thinned events, $\{\tilde{x}_m\}_{m=1}^M$. 3) Resampling the intensity function, $\mathbf{g}_{\mathbf{M}+\mathbf{K}}$.

Metropolis-Hastings is used to sample M . The probability of insertion/deletion is parameterised by a Bernoulli proposal function: $b(K, M) : \mathbb{N} \times \mathbb{N} \rightarrow (0, 1)$, where the parameter has been arbitrarily set to $\frac{1}{2}$. If an insertion is required, a new x_{M+1} is drawn uniformly and at random from $\mu(\mathcal{T})$, and $g(x_{M+1})$ is drawn from the Gaussian process conditioned on the current state. A deletion results in a thinned event \tilde{x}_m being removed at random from $\{\tilde{x}_m\}_{m=1}^M$. The overall transition kernels q , and Metropolis-Hastings acceptance ratios, a , are:

$$q_{ins}(M+1 \leftarrow M) = \frac{b(K, M)}{\mu(\mathcal{T})} \mathcal{GP}(g(\tilde{x}_{M+1}) \mid \{\tilde{x}_m\}_{m=1}^M, \mathbf{g}_{M+K}), \quad (15)$$

$$a_{ins} = \frac{(1 - b(K, M+1))\mu(\mathcal{T})\lambda^*}{(M+1)b(K, M)(1 + \exp(g(\tilde{x}_{M+1})))}, \quad (16)$$

$$q_{del}(M-1 \leftarrow M) = \frac{1 - b(K, M)}{M} \quad (17)$$

$$a_{del} = \frac{Mb(K, M-1)(1 + \exp(g(\tilde{x}_m)))}{(1 - b(K, M))\mu(\mathcal{T})\lambda^*}. \quad (18)$$

Sampling the locations of the thinned events also makes use of the Metropolis criterion. For each event \tilde{x}_m a move to \hat{x}_m is proposed via a Gaussian proposal density. A function value $g(\hat{x}_m)$ is then drawn conditioned on the state with $g(\tilde{x}_m)$ removed, denoted \mathbf{g}_{M+K} . This gives the move acceptance ratio

$$a_{move} = \frac{q_{move}(\tilde{x}_m \leftarrow \hat{x}_m)(1 + \exp(g(\tilde{x}_m)))}{q_{move}(\hat{x}_m \leftarrow \tilde{x}_m)(1 + \exp(g(\hat{x}_m)))}. \quad (19)$$

where q_{move} is the proposal distribution. We use a symmetric Gaussian proposal

$$q_{move}(\hat{x}_m \leftarrow \tilde{x}_m) = \mathcal{N}\left(0, \frac{\mu(\mathcal{T})}{100}\right). \quad (20)$$

To sample the function we opt to use Elliptical Slice Sampling (Murray et al., 2010). This is an algorithm specifically designed for sampling from high dimensional, highly correlated, Gaussian process posteriors. The log conditional posterior over function values is

$$\begin{aligned} \ln p(\mathbf{g}_{M+K} \mid M, \{x_k\}_{k=1}^K, \{\tilde{x}_m\}_{m=1}^M, \gamma) = \\ -\frac{1}{2}\mathbf{g}_{M+K}\mathbf{\Sigma}^{-1}\mathbf{g}_{M+K} - \sum_{k=1}^K \ln(1 + \exp(-g(x_k))) \\ - \sum_{m=1}^M \ln(1 + \exp(g(\tilde{x}_m))) + \text{const.} \end{aligned} \quad (21)$$

In our case, $\mathbf{\Sigma}$ is equal to the covariance in Equation 11, and naturally for each iteration we perform all the above updates in parallel for each observed point process, conditioned on the latent functions.

To infer the posteriors over the Gaussian process hyperparameters, we use Hamiltonian Monte Carlo

Algorithm 1 MCMC Scheme

Input: $\{X_k\}_{k=1}^K$, priors.

repeat

ParFor $d = 1$ to D

 Sample thinned events: Equations 16 \rightarrow 27

 Sample locations: Equations 20 \rightarrow 19

 Sample function: Equation 21

 Sample hyperparameters: Equation 21

 Sample λ^* : Equation 22

EndParFor

 Sample latent functions: Equation 13

 Sample latent hyperparameter: Equation 23

until *convergence* is *true*

(HMC) (Duane et al., 1987; Neal, 2010), with log-normal priors over each hyperparameter. By placing a Gamma prior with shape α and inverse scale β over λ^* , we infer the posterior conditioned on the thinned and true points using a Gibbs update as follows:

$$\alpha_{post} = \alpha + K + M, \quad \beta_{post} = \beta + \mu\mathcal{T}. \quad (22)$$

3.2 LEARNING THE LATENT FUNCTIONS

Conditioning on the point process intensity functions, $g_d(x)$, the latent functions are dependent, with conditional posterior distribution given by Equation 13.

Having drawn new values for each of the $u_q(Z_q)$, we can update the ϕ_q using a metropolis-hastings step under the following log conditional posterior which is

$$\begin{aligned} \ln p(\phi_q \mid u_q(Z_q), Z_q) = & -\frac{1}{2}u_q(Z_q)\mathbf{K}_{u_q, u_q}^{-1}\frac{1}{2}u_q(Z_q) \\ & -\frac{1}{2}\log \det(\mathbf{K}_{u_q, u_q}) + \text{const.} \end{aligned} \quad (23)$$

The overall procedure is summarised in algorithm 1.

4 ADAPTIVE THINNING

In higher dimensional spaces, data is typically concentrated into small, high density sub-domains. Under the current methodology, we must thin the entire empty space to a uniform concentration which matches that of the most dense subregion. If we wish to use Gaussian Process intensities this rapidly becomes infeasible, even under the most radical of sparse approximations (Snelson and Ghahramani, 2005).

Our novel solution to this problem is to model the upper bounded intensity over the space using a piecewise constant function, where each section takes a fractional proportion of the global upper bound, λ^* . This preserves the tractability of the integrals in the likelihood and posterior, and does not violate any of the properties of the point process, while simultaneously

allowing empty regions to be thinned to a far lower average density.

Consider Figure 1: this shows both the data and the thinned points, where for the left three quarters of the plot the maximum rate does not exceed 50% of λ^* . Let us assume we allow the maximum rate to take one of two values for each datapoint: $\frac{1}{2}\lambda^*$ and λ^* . For each new thinned point we sample an intensity function value, before also sampling an upper bound for the rate from the available levels. This upper bound is at least as great as the current function evaluation at that point.

In this manner we hope to infer that for the majority of Figure 1, the rate may be happily upper-bounded by half the global maximum rate, λ^* , and hence the bulk of the space may be thinned to a significantly lower density. As a result, the computational burden incurred will be significantly reduced, as far fewer expensive points need be incorporated into our GP.

In our particular implementation, we fix *a-priori* a set of B possible maximum rate ‘levels’:

$$L = \{l_i \in (0, 1] | l_i < l_{i+1}, l_B = 1\}_{i=1}^B. \quad (24)$$

We then augment the variable set to include for each thinned point \tilde{x}_m which rate level $r_m \in \{1 \dots B\}$ it is currently assigned, where we set r_m such that $\sigma(g(\tilde{x}_m)) \leq l_{r_m}$. This causes the probability of seeing a thinned point \tilde{x}_m under the sigmoid GP to become

$$p(\tilde{x}_m | r_m) = \frac{l_{r_m} - \sigma(g(\tilde{x}_m))}{l_{r_m}}, \quad (25)$$

while the probability of a non-thinned point remains unchanged. Using this relationship we modify the Metropolis acceptance criteria which now become

$$a_{ins} = \frac{(1 - b(K, M + 1))\mu(\mathcal{T})\lambda^* l_{r_{M+1}} p(\tilde{x}_{M+1} | r_{M+1})}{(M + 1)b(K, M)}, \quad (26)$$

$$a_{del} = \frac{Mb(K, M - 1)}{(1 - b(K, M))\mu(\mathcal{T})\lambda^* l_{r_m} p(\tilde{x}_m | r_m)}, \quad (27)$$

as well as the likelihood function for $p(g(\mathbf{X}_{M+K}))$, Equation 21.

In principle this scheme could slow mixing, since the function is constrained to lie below the maximum level at each point. By ensuring that there is always some slack, s , between the function and the rate level assigned we find that mixing is hardly affected. The slack is incorporated by assigning the rate as follows:

$$r_m \leftarrow \begin{cases} \underset{r}{\operatorname{argmax}} \{ \sigma(g(\tilde{x}_m)) \leq l_r \times s \}, & \sigma(g(\tilde{x}_m)) \leq s \\ 1, & \text{otherwise.} \end{cases} \quad (28)$$

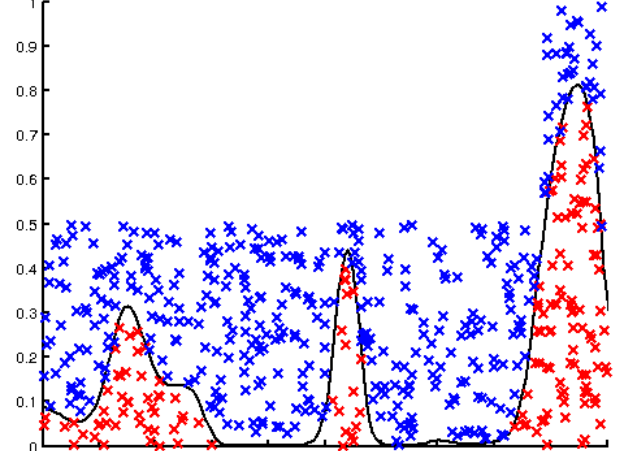


Figure 1: Graphical representation of adaptive thinning: Blue crosses indicate thinned points, red crosses represent data. The black line shows the intensity function. Each point is accepted as data with probability given by $\sigma(g(x_n))$. Fewer thinned points are required in areas of half maximum bound.

We used $s = 0.9$. The rate levels can also change at each iteration during the ‘move’ step when we compute the new rate level for jittered points and we compose the acceptance criteria as the product of the insertion and deletion criteria $a_{move} = a_{ins} \times a_{del}$.

Finally, when re-sampling λ^* we must compute an estimate of the total number of points under a single rate uniformisation of the space. This estimate is readily available because the number of points (including observed data points) with rate r is a Monte-Carlo integral of the proportion of space thinned to rate level l_r . Since the number of points in each region is scaled by l_r , the estimated total is

$$\hat{N}_{tot} = \sum_{k=1}^K \frac{1}{l_{\bar{r}_k}} + \sum_{m=1}^M \frac{1}{l_{r_m}}, \quad (29)$$

where \bar{r}_k is the notional rate of observed data computed exactly as for the thinned points. The posterior value α_{post} is therefore $\alpha + \hat{N}_{tot}$.

To validate adaptive thinning, we return to the original SGCP and modify it in the manner described above. We perform two experiments: The first in 1D and the second in 2D. In both cases we use 10 known random intensity functions to generate event data: In the 1D case we sample 15 random datasets per function, while in the 2D case we generate 10. One dataset is used to learn the model, the rest are held out for testing purposes. Two metrics of performance are used: L2-norm error as measured against the true intensity function, and average predictive log-likelihood across all held

out test datasets.

In 1D, we run each model for 6 minutes total compute time, in 2D we allow 20 minutes total. In both cases half the time is allocated to burn-in. In 1D the single rate method achieved roughly one sample per second, with the two rate case yielding just under four samples per second, and the four rate approach giving just under 8 samples per second. In 2D the number of points required was larger in all cases, with the multi-rate approach buying a factor of two speedup.

The results, (given in Tables 1 through 4), show that in almost all cases the multi-level approach performs best across both metrics. It is observed that typically the original, homogenous rate approach performs worst of all.

5 EMPIRICAL RESULTS

As this is the first model for structured point process data, the approach is initially validated on a synthetic dataset. It is then compared to both the independent SGCP, as well as a state of the art Kernel Density Estimator (Botev et al., 2010) on two real datasets.

5.1 SYNTHETIC DATA

Using the convolution process, we sample four intensity functions, using those to sample event data. The variety of intensities which may be observed given a single latent function is notable in Figure 2.

We then average over 2000 iterations after it was determined convergence had been achieved. The resulting learned intensity functions are shown in Figure 3.

It is reassuring that the original latent function is well recovered given only four observed event processes.

5.2 REAL DATA

Two datasets were selected to test the model, both of which we considered were likely to exhibit a dependency structure which could be well captured by the convolution process.

- British politicians (MPs) tweet times during the week of Nelson Mandela’s death (02/12/13–08/12/13). These were obtained using the Twitter API. Here we considered that there would naturally be a daily periodicity, however, it is not unreasonable to further postulate that some MPs may concentrate their twitter activity into a smaller segment of the day. This behaviour should be well captured by the convolution process.
- NBA player shot profiles for the 2013-2014 season, scraped from the NBA website. Here we se-

lect a diverse subset of four players: Blake Griffin, Damien Lillard, DeMar DeRozan, and Arron Affalo. It was supposed that it might be possible for a single latent function to be blurred to represent a variety of player positions and styles.

5.2.1 Twitter Data Results

Four MPs active on twitter were selected at random. Here on we call them MPs A, B, C, and D. We select data from the period covering 02/12/13 through 13/12/13, and randomly partition each dataset into 75% training data, with the remainder being used to evaluate predictive test log-likelihood.

Figure 4 depicts the average learned intensity functions for each MP (red line), along with the one standard deviation bars (grey shading) derived from the function samples. The bottom plot depicts the learned latent driving function in the same manner.

The latent function clearly shows a strong daily period, particularly evident during the working week (02/12/13 was a Monday—corresponding to ‘1’ in Figure 4). Furthermore, the largest two peaks in activity occur on the 3rd and the 5th of December. Potential contributing factors to these two spikes include a public sector strike, and the death of Nelson Mandela respectively.

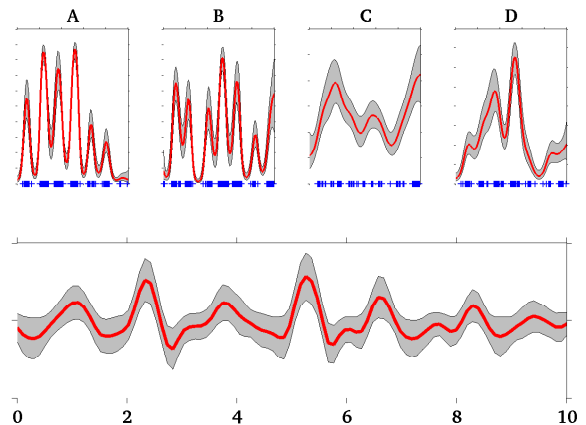


Figure 4: Learned intensities over four MP’s tweet data (A, B, C, D); learned latent function at bottom. Actual data shown in blue.

Table 5 gives predictive log-likelihood for the held out data, again evaluated across three approaches: An intensity function learned via Kernel Density Estimation (KDE) (Botev et al., 2010), the SGCP (Adams et al., 2009), and our own structured approach. Both the SGCP and our own approach use two maximum rate levels for adaptive thinning. Each intensity function is modelled using an independent SGCP/KDE. The

Table 1: One dimensional adaptive thinning
L2-norm function error. Bold is best.

Function (1D)	L2 Norm Error		
	Original	2 Rates	4 Rates
1	11.6	13.0	7.1
2	14.6	10.5	7.2
3	10.6	5.0	4.9
4	10.5	4.7	5.1
5	12.4	10.1	10.4
6	11.1	8.2	7.6
7	12.0	13.7	12.5
8	13.0	12.0	12.8
9	19.6	16.4	28.8
10	31.4	27.2	32.6

Table 2: One dimensional adaptive thinning
average predictive log-likelihood on 14 held out
datasets. Bold is best.

Function (1D)	Predictive Log-Likelihood		
	Original	2 Rates	4 Rates
1	373.8	381.8	388.2
2	626.1	644.2	650.7
3	274.2	285.1	288.0
4	435.5	457.7	456.0
5	877.0	885.8	889.5
6	995.4	1006.6	1013.4
7	753.0	763.3	760.6
8	522.3	531.2	528.3
9	1840.6	1852.9	1826.0
10	2328.1	2365.8	2349.8

Table 3: Two dimensional adaptive thinning
L2-norm function error. Bold is best.

Function (2D)	L2 Norm Error		
	Original	2 Rates	4 Rates
1	13.3	13.4	11.3
2	14.3	14.3	14.7
3	13.5	14.7	13.5
4	12.5	12.9	12.0
5	17.9	16.6	17.8
6	14.4	16.2	15.3
7	15.1	13.7	17.5
8	14.6	14.4	14.8
9	18.3	17.1	15.6
10	15.4	12.9	13.6

Table 4: Two dimensional adaptive thinning
average predictive log-likelihood on 9 held out
datasets. Bold is best.

Function (2D)	Predictive Log-Likelihood		
	Original	2 Rates	4 Rates
1	2039.6	2080.6	2203.0
2	2757.9	2758.1	2829.3
3	2753.2	2689.5	2827.2
4	2803.2	2784.0	2933.6
5	2532.4	2663.0	2572.1
6	3098.2	3040.5	3054.4
7	3157.5	3259.8	3075.2
8	2086.9	2101.0	2087.4
9	5018.1	5146.6	5185.3
10	2008.1	2205.0	2174.0

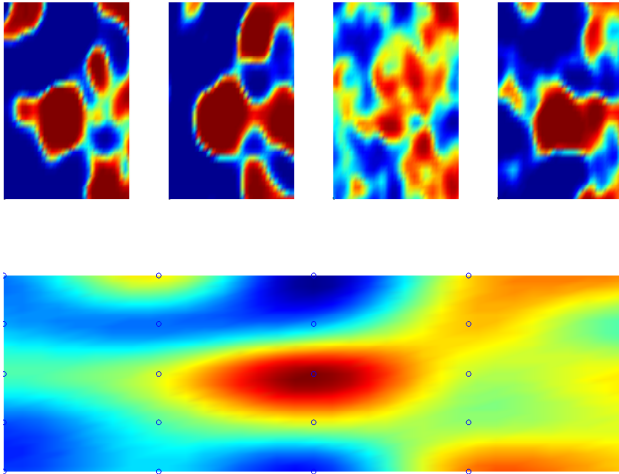


Figure 2: Synthetic functions (not showing the
sampled events).

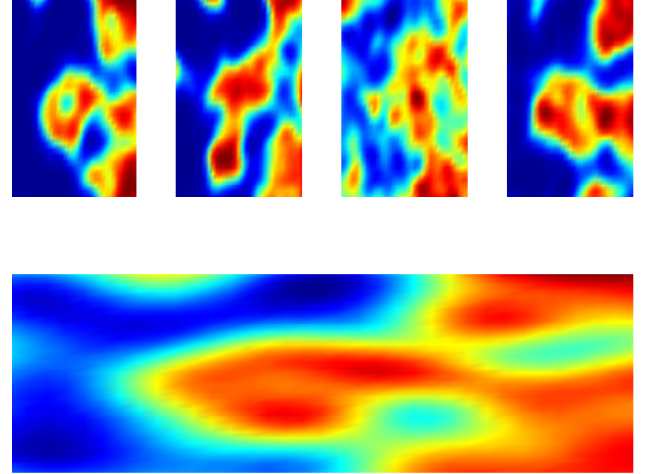


Figure 3: Learned functions using 3 rate levels: $\frac{1}{4}\lambda^*$,
 $\frac{1}{2}\lambda^*$, λ^* .

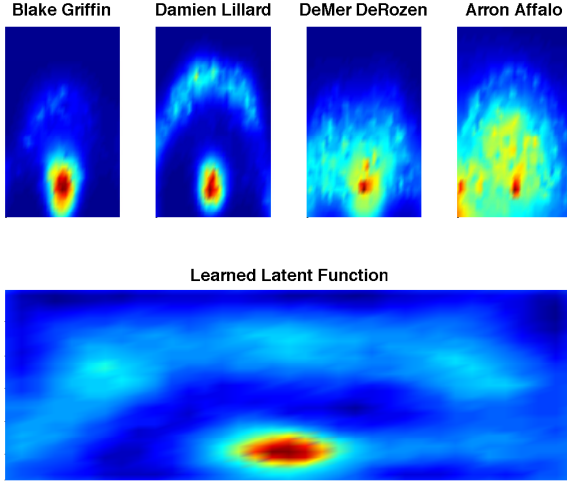


Figure 5: Learned basketball intensity functions using 3 rate levels: $\frac{1}{4}\lambda^*$, $\frac{1}{2}\lambda^*$, and λ^* .

structured approach performs vastly better, suggesting that it is highly appropriate for this type of data.

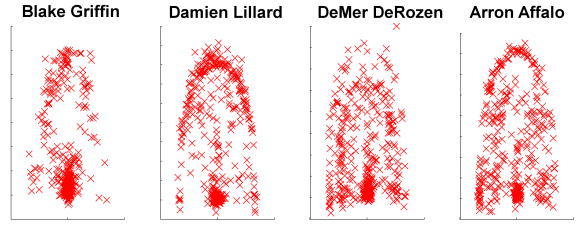
Table 5: Predictive log-likelihood for held out twitter data across models.

MP	Predictive Log-Likelihood		
	KDE	SGCP	Ours
A	177.1	176.6	469.5
B	-1.3	89.6	412.9
C	-5.4	49.6	283.4
D	-67.7	38.7	293.7

5.2.2 Basketball Data

For the basketball point shot data, the approach performed particularly well. Each player had around 600 attempted shots, of which we used 400, holding out the rest as test data. We used 3 rate boundaries: $\frac{1}{4}\lambda^*$, $\frac{1}{2}\lambda^*$, λ^* , and once again averaged over 2000 samples after convergence. We compare predictive log-likelihood to both the SGCP model (using the same set of rate boundaries) and using a rate function estimated via a state of the art KDE by Botev et al. (2010).

Figure 5 depicts the resulting intensity functions. It is clear that the latent function represents a general view of the court hotspots, the hoop and three pointer line are clearly demarcated. Furthermore the intensity functions for each player strongly match what would be expected given their playing style—e.g. Arron Affalo is a ‘shooting guard’ who is expected to spend the majority of his time inside the three pointer line,



Player	Predictive Log-Likelihood		
	KDE	SGCP	Ours
Blake Griffin	-121.8	335.6	374.7
Damien Lillard	-22.6	231.2	395.3
DeMer DeRozen	-2.9	253.1	410.7
Arron Affalo	-260.7	-76.7	84.2

Figure 6: Basketball ball shot data (top) and predictive log-likelihood for held out basketball data across models (bottom).

but has a propensity to shoot from the bottom left of the court. These effects are clearly visible on the heat map, less so on the data (see Figure 5.2.1).

As is clearly demonstrated in Table 6, our structured approach to modelling the basketball point data in a fully Bayesian fashion yields a huge improvement over both the independent SGCP as well as a modern kernel density estimator. Another point worth making is that due to the high data density around the hoop for each player, the traditional approach of thinning (used here as well as in the SGCP) would be prohibitively computationally expensive. We are only able to test on this data due to the method of adaptive thinning introduced in this paper.

6 CONCLUSION

We have introduced a fully generative model for dependent point processes, alongside an efficient, parallelised inference scheme. We have shown the appropriateness of this model on two real datasets, and introduced a new adaptation of thinning which allows the model to scale to larger datasets and in particular higher dimensional spaces. Future work entails investigating the appropriateness of manually introducing known latent drivers, exploring multiple latent functions, and replacing the MCMC inference scheme with one based on stochastic variational inference (Hensman et al., 2013).

Acknowledgements

Tom Gunter is supported by UK Research Councils. Chris Lloyd is funded by a DSTL PhD Studentship.

References

- Ryan Prescott Adams, Iain Murray, and David J. C. MacKay. Tractable Nonparametric Bayesian Inference in Poisson Processes with Gaussian Process Intensities. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09*, pages 9–16, New York, NY, USA, 2009. ACM.
- Mauricio A. Álvarez and Neil D. Lawrence. Computationally Efficient Convolved Multiple Output Gaussian Processes. *J. Mach. Learn. Res.*, 12:1459–1500, July 2011. ISSN 1532-4435.
- Mauricio A. Alvarez, David Luengo, and Neil D. Lawrence. Latent Force Models. In *International Conference on Artificial Intelligence and Statistics*, pages 9–16, 2009.
- ZI Botev, JF Grotowski, and DP Kroese. Kernel Density Estimation via Diffusion. *The Annals of Statistics*, 38(5):2916–2957, 2010.
- Simon Duane, A. D. Kennedy, Brian J. Pendleton, and Duncan Roweth. Hybrid Monte Carlo. *Physics Letters B*, 195(2):216 – 222, 1987. ISSN 0370-2693.
- James Hensman, Nicolás Fusi, and Neil D. Lawrence. Gaussian Processes for Big Data. *CoRR*, abs/1309.6835, 2013.
- J. F. C. Kingman. *Poisson Processes (Oxford Studies in Probability)*. Oxford University Press, January 1993. ISBN 0198536933.
- P. A. W. & Shedler G.S. Lewis. Simulation of a Non-homogeneous Poisson Process by Thinning. *Naval Research Logistics Quarterly*, 26:403–413, 1979.
- Iain Murray, Ryan P. Adams, and David J. C. MacKay. Elliptical Slice Sampling. *Journal of Machine Learning Research - Proceedings Track*, 9:541–548, 2010.
- R. M. Neal. *Handbook of Markov Chain Monte Carlo*. Chapman and Hall/CRC Press, 2010.
- Natesh S Pillai, Qiang Wu, Feng Liang, Sayan Mukherjee, and Robert L Wolpert. Characterizing the function space for bayesian kernel models. *Journal of Machine Learning Research*, 8(8), 2007.
- Joaquin Quiñero Candela and Carl Edward Rasmussen. A Unifying View of Sparse Approximate Gaussian Process Regression. *J. Mach. Learn. Res.*, 6:1939–1959, December 2005. ISSN 1532-4435.
- Vinayak Rao and Yee Whye Teh. Gaussian Process Modulated Renewal Processes. In *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011, Granada, Spain*, pages 2474–2482, 2011.
- Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2nd edition 2006 edition, 2006. ISBN 0-262-18253-X.
- E. Snelson and Z. Ghahramani. Sparse Gaussian Processes using Pseudo-inputs. In *NIPS 18*, 2005.
- Y. W. Teh, M. Seeger, and M. I. Jordan. Semiparametric Latent Factor Models. In *Proceedings of the International Workshop on Artificial Intelligence and Statistics*, volume 10, 2005.

Learning Peptide-Spectrum Alignment Models for Tandem Mass Spectrometry

John T. Halloran

Dept. of Electrical Engineering
University of Washington
Seattle, WA 98195, USA

Jeff A. Bilmes

Dept. of Electrical Engineering
University of Washington
Seattle, WA 98195, USA

William S. Noble

Department of Genome Sciences
University of Washington
Seattle, WA 98195

Abstract

We present a peptide-spectrum alignment strategy that employs a dynamic Bayesian network (DBN) for the identification of spectra produced by tandem mass spectrometry (MS/MS). Our method is fundamentally generative in that it models peptide fragmentation in MS/MS as a physical process. The model traverses an observed MS/MS spectrum and a peptide-based theoretical spectrum to calculate the best alignment between the two spectra. Unlike all existing state-of-the-art methods for spectrum identification that we are aware of, our method can learn alignment probabilities given a dataset of high-quality peptide-spectrum pairs. The method, moreover, accounts for noise peaks and absent theoretical peaks in the observed spectrum. We demonstrate that our method outperforms, on a majority of datasets, several widely used, state-of-the-art database search tools for spectrum identification. Furthermore, the proposed approach provides an extensible framework for MS/MS analysis and provides useful information that is not produced by other methods, thanks to its generative structure.

1 INTRODUCTION

A fundamental problem in biology and medicine is accurately identifying the proteins present in a complex sample, such as a drop of blood. The only high-throughput method for solving this problem is *tandem mass spectrometry* (MS/MS). Given a complex sample, an MS/MS experiment produces a collection of spectra, each of which represents a single *peptide* (protein subsequence) that was present in the original sample. Fundamental to MS/MS is the ability to accurately identify the peptide responsible for generating a particular spectrum.

The most accurate methods for identifying MS/MS spectra make use of a peptide database. Given a peptide drawn from

the database and an observed spectrum, these methods compare a *theoretical spectrum* of the peptide’s idealized fragmentation events to a quantized or fixed-width thresholded observed spectrum. Such preprocessing necessarily discards potentially useful information. The spectrum identification problem is greatly complicated by experimental noise, corresponding both to the presence of unexpected peaks (insertions) and the absence of expected peaks (deletions) in the observed spectrum (Fig. 1). This paper describes a Dynamic Bayesian network for Rapid Identification of Peptides (DRIP), a database search method that serves as a generative model of the process by which peptides produce spectra in MS/MS. DRIP explicitly models insertions and deletions, without quantization or thresholding of the observed spectra.

We note that a DBN-based database search method, called Didea, was recently proposed [1], but this method does not model the underlying process by which peptides produce MS/MS spectra. Rather, in Didea both theoretical and observed spectra are observed, and the model contains only a single hidden variable, which is devoid of any physical meaning relative to the underlying MS/MS process. The theoretical spectrum in DRIP, by contrast, is hidden; insertions and deletions are explicitly modeled as latent variables (as in [2]), and the most probable alignment between the theoretical and observed spectra can be efficiently calculated (detailed in Section 4). Furthermore, Didea has a single hyperparameter that is optimized via grid search, making the model poorly adaptable to the wide range of machines with widely varying characteristics, a problem addressed by the highly trainable nature of DRIP.

We demonstrate, in fact, that against four state-of-the-art benchmarked competitors, DRIP is the most frequent top performer, dominating the others on four out of nine separate datasets. By contrast, other competitors, such as Didea, dominate on at most two datasets. Furthermore, DRIP, thanks to its generative approach, provides valuable auxiliary information, such as which observed peaks are most likely spurious, which theoretical peaks are most likely present, and the ability to calculate posteriors of interest via sum-product inference [3, 4]. Such posteriors include the probability of post-translational modifications given the observed spec-

trum, a task which previously required post-processing the results of a database search [5].

We first give a brief overview of a typical tandem mass spectrometry experiment and an overview of database search in Section 2. Readers are directed to [6] for further background in this area. Next, the four benchmarked competitors are described in Section 3. DRIP is described in detail in Section 4. Results are presented in Section 5, and we conclude and discuss future work in Section 6.

2 TANDEM MASS SPECTROMETRY AND DATABASE SEARCH

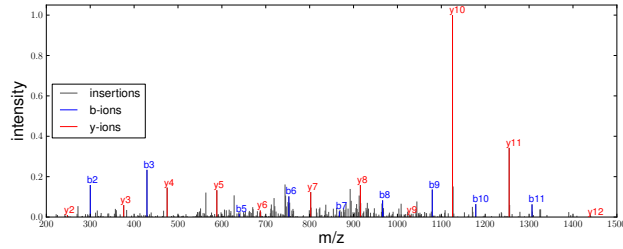


Figure 1: Sample tandem mass spectrum, where the peptide responsible for generating the spectrum is $x = \text{LWEPLLDVVLVQTK}$, the precursor charge c^s is 2, and the most probable alignment computed in DRIP is plotted. The b-ion peaks are colored blue, y-ion peaks are colored red, and insertions are colored gray. Note that fragment ions b_1, y_1, b_4, b_{12} correspond to deletions.

Although we are typically interested in the protein content of a complex mixture, the fundamental unit of observation in tandem mass spectrometry is the peptide, because peptides are more amenable to liquid chromatography and mass spectrometry analysis[6, 7]. Thus, a typical MS/MS experiment begins by digesting the proteins into peptides using a cleavage agent such as the enzyme trypsin. MS/MS then proceeds with two rounds of mass spectrometry. The first round measures the mass-to-charge ratio (m/z) of the intact peptide (called the *precursor m/z*), and the second round fragments the peptide and measures the m/z values of the resulting prefixes and suffixes. Each of these fragment m/z values is associated with an intensity value, which is roughly proportional to the number of copies of that peptide fragment. Figure 1 displays a sample tandem mass spectrum, along with the theoretical fragment ions (described below) of the generating peptide. A single unit along the m/z axis is called a *Thomson* (Th), and the intensity (y-axis) is unitless but can be seen as a measure of abundance or count.

Let \mathbb{P} be the set of all possible peptides and S be the set of all tandem mass spectra. Given an observed spectrum $s \in S$ with observed precursor m/z m^s and precursor charge c^s , our task is to identify the peptide x from a given peptide database $\mathcal{D} \subseteq \mathbb{P}$ that is responsible for generating s . Any given mass spectrometry device is capable of isolating

peptides with a specified precision on the precursor m/z ; therefore, we may constrain the search to only consider peptides with precursor $m/z \pm w$ of m^s . The set of *candidate peptides* to be scored is then

$$D(m^s, c^s, \mathcal{D}, w) = \left\{ x : x \in \mathcal{D}, \left| \frac{m(x)}{c^s} - m^s \right| \leq w \right\}, \quad (1)$$

where $m(x)$ is the calculated mass of peptide x . The goal of database search, then, is to return the highest scoring candidate peptide

$$x^* = \underset{x \in D(m^s, c^s, \mathcal{D}, w)}{\operatorname{argmax}} \psi(x, s),$$

where $\psi : \mathbb{P} \times S \rightarrow \mathbb{R}$ is a function that assigns higher scores to higher quality matches and the pair (x, S) is referred to as a *peptide-spectrum match* (PSM). The primary distinguishing characteristic of any database search procedure is its choice of score function ψ .

2.1 THEORETICAL SPECTRA

Many score functions, including the one employed by the very first database search algorithm, SEQUEST [8], work by comparing the observed spectrum to a *theoretical spectrum* that is derived from a candidate peptide using basic rules of biochemistry. Let $x \in D(m^s, c^s, \mathcal{D}, w)$ be an arbitrary candidate peptide of length n . Note that $x = x_0x_1 \dots x_{n-1}$ is a string of *amino acids*, i.e. characters in a dictionary of size 20. For convenience, let $\tilde{n} = n - 1$. Our goal is to produce a theoretical spectrum v^x containing the fragment m/z values that we expect x to produce. In this work, we assume that the mass spectrometer employs collision-induced dissociation, which is the most widely employed method of peptide fragmentation. The model can be modified in a straightforward fashion to accommodate other fragmentation modes.

The first type of fragment m/z value corresponds to prefixes or suffixes of the candidate peptide, referred to respectively as *b-ions* and *y-ions*. In this work, we assume that the precursor charge c^s is 2, because this is the charge state of the high-quality set of PSMs used for training [9]. For $c^s = 2$, these b- and y-ions can be represented as functions $b(\cdot, \cdot)$ and $y(\cdot, \cdot)$, respectively, that take as input a peptide x and integer $k < n$:

$$b(x, k) = \sum_{i=0}^{k-1} m(x_i) + 1, \quad y(x, k) = \sum_{i=\tilde{n}-k}^{\tilde{n}} m(x_i) + 19. \quad (2)$$

Note that the whole peptide mass is not considered and that, for $1 < k < n$, we have the recurrence relations $b(x, k) = b(x, k-1) + m(x_{k-1})$ and $y(x, k) = y(x, k-1) + m(x_{\tilde{n}-k})$. In Equation 2, the b-ion unit offset corresponds to the mass of a hydrogen atom while the y-ion offset corresponds to the masses of a water molecule as well as a hydrogen atom.

Thus, the b- and y-ions are simply the shifted prefix sums and suffix sums of x , respectively. When there is no ambiguity as to the peptide being described, it is typical to represent the b- and y-ion pairs as (b_k, y_{n-k}) for $k = 1, \dots, \tilde{n}$, where the subscript denotes the number of amino acids utilized in the ion computation. As an example, for peptide $x = \text{EALK}$, $(b_1, y_3) = (b(x, 1), y(x, 3)) = (130, 331)$. Denoting the number of unique b- and y-ions as n^x and, for convenience, letting $\tilde{n}^x = n^x - 1$, our theoretical spectrum is a sorted vector $v^x = (v_0, \dots, v_{\tilde{n}^x})$ consisting of the unique b- and y-ions of x . Figure 2 displays the theoretical spectrum for $x = \text{EALK}$, and Figure 1 displays an observed spectrum with annotated b- and y-ions.

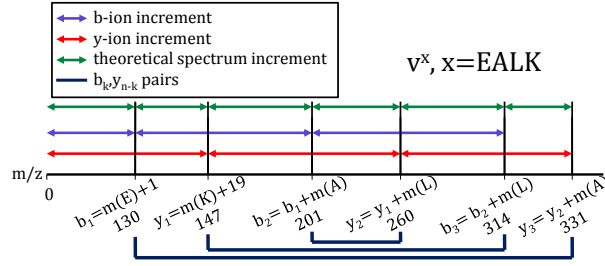


Figure 2: Theoretical spectrum of the peptide $x = \text{EALK}$. Note that the b- and y-ions correspond to prefix and suffix sums, respectively, of the peptide x .

3 PREVIOUS WORK

We compare DRIP’s performance to that of four previously developed state-of-the-art methods. We describe each method briefly here, and in more detail in [10]. All four methods begin by binning the observed spectrum. The first database search algorithm, SEQUEST [8], uses a scoring function called XCorr, consisting of a dot-product minus a cross-correlation term that provides an empirical null model for a random match. The second approach, the Open Mass Spectrometry Search Algorithm (OMSSA) [11] counts, the b- and y-ions present in the observed spectrum and then estimates a p-value by fitting this count to a Poisson distribution with mean parameter derived from the properties of the observed spectrum. The third algorithm, MasS Generating Function DataBase (MS-GFDB) [12], computes a score by taking a dot product between a Boolean theoretical vector and a processed observed spectrum and then computes a p-value for this score using dynamic programming. The fourth algorithm that we consider, Didea [1], is most closely related to DRIP, in the sense that both methods employ a DBN. However, Didea differs from DRIP in four quite significant ways:

- **Notion of “time.”** In Didea, each frame of the DBN corresponds to one amino acid from the candidate peptide sequence. Accordingly, Didea must copy the entire observed spectrum in every frame in order to score these observations. By contrast, each frame in DRIP

instead corresponds to a peak in the observed spectrum, such that a single m/z value and intensity value are observed and scored per frame.

- **Whether the theoretical spectrum is hidden or observed.** The theoretical spectrum in DRIP is hidden, and inference is run to determine the best alignment between the observed and theoretical spectra while accounting for insertions and deletions, thus providing not just a score but valuable alignment information as well. In Didea, the theoretical spectrum is not hidden because the amino acid variables in each frame are observed, so that performing inference only provides a score.
- **Observed spectrum pre-processing.** DRIP performs much less pre-processing on the observed spectrum than Didea. In particular, Didea must work with a version of the observed spectrum in which the m/z axis is discretized and the observed intensity values are reweighted using a complicated function of exponentials. DRIP instead scores m/z values in their natural resolution, without discarding information due to quantization.
- **Training of parameters.** Whereas Didea is essentially a fixed model, DRIP offers the ability to learn its parameters using training data. The only learning available in Didea is the tuning of a single hyperparameter, via grid search, which controls the reweighting of peak intensities.

4 DRIP PEPTIDE SCORING

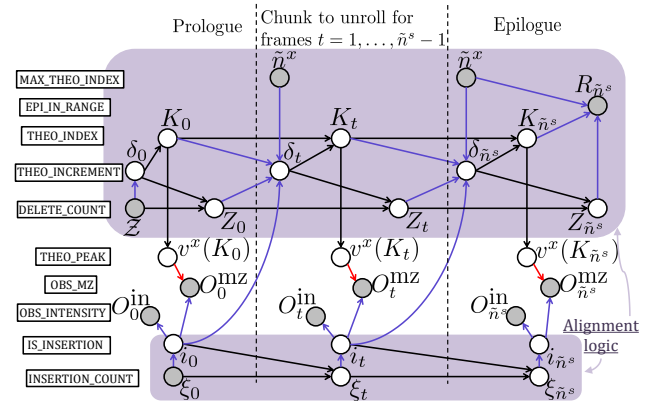


Figure 3: Graph of DRIP, where the boxed words on the far left summarize the role of the random variables on the same horizontal level to aid interpretation of the model. THEO and OBS are short for theoretical and observed, respectively.

The graph of DRIP is displayed in Figure 3, where each frame of the model corresponds to a single observed peak. Shaded nodes represent observed variables, and unshaded nodes represent hidden variables. Black edges correspond to deterministic functions of parent variables, and blue edges

represent switching parent functionality (also known as Bayesian multi-nets [13]) where the parent nodes are known as *switching parents* and the parents (and hence conditional distributions) of their children may change given the values of their switching parents. Finally, red edges denote continuous conditional distributions. Random variables are grouped into frames, indexed by $t = 0, \dots, n^s - 1$. Note that while elements of vectors are denoted using parentheses, a particular value of a sequence is denoted using subscripts, such that δ_t is a random variable in the t th frame. For convenience, let $\tilde{n}^s = n^s - 1$, and recall that \tilde{n}^x denotes the number of peaks in the theoretical spectrum minus one. The first and last frames are known as the *prologue* and *epilogue*, respectively. The middle frame is called the *chunk* and is unrolled $n^s - 2$ times to frames $t = 1, \dots, \tilde{n}^s - 1$. Each frame of the graph contains observations $O_t^{\text{m/z}}$ and O_t^{int} , the t th m/z and normalized intensity values of s , respectively. Thus, we can view traversing the graph from left to right as moving across the observed spectrum.

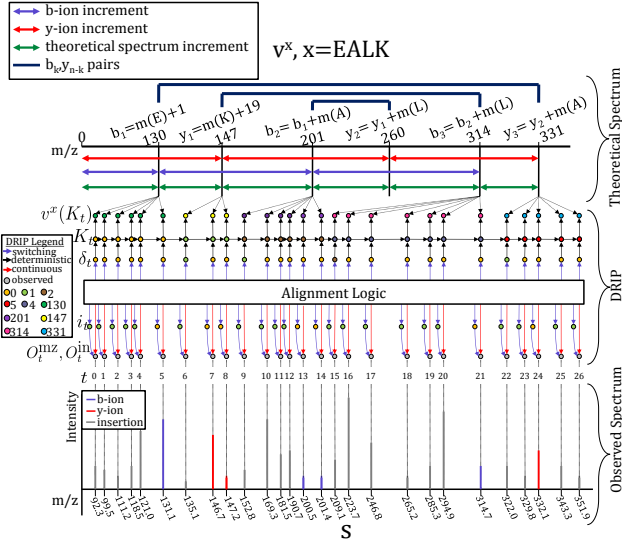


Figure 4: Illustration of a particular spectrum alignment (instantiation of random variables) in DRIP, where the node color denotes its instantiated value. The observed spectrum peaks serve as the observations in each frame while the theoretical spectrum is hidden. DRIP thus serves as a sequencer through all possible alignments between the theoretical and observed spectra, made efficient via dynamic programming.

The goal of DRIP is to calculate the most probable alignment between the observed and theoretical spectra, where an alignment is an instantiation of the random variables in the graph and the scoring of observed peaks as dictated by this instantiation. This concept is detailed in Figure 4 for a particular alignment, where random variable values are denoted by node colors, and the alignment corresponds to a traversal of both the theoretical (upper portion) and observed spectra (lower portion). In the center portion of Figure 4, the random variable K_t denotes the theoretical peak index and, given the increment random variable δ_t , moves us down the

theoretical spectrum, while further alignment logic in DRIP constrains the manner in which the theoretical and observed spectra may be aligned. Figure 5 illustrates the scoring of observed peaks (discussed in Section 4.2) in this alignment, and the instantiation of random variables may be found in Table 1. We now discuss the details of how DRIP aligns the theoretical and observed spectra.

4.1 TRAVERSING THE THEORETICAL SPECTRUM

The variable K_t is the index of the theoretical peak used to score peaks in frame t , such that

$$p(K_0 = \delta_0 | \delta_0) = 1, \quad (3)$$

$$p(K_t = K_{t-1} + \delta_t | K_{t-1}, \delta_t) = 1, t > 0. \quad (4)$$

From (3) and (4), we see that δ_t is the number of theoretical peaks we traverse between frames t and $t + 1$. Note that a deletion thus occurs when $\delta_0 > 0$ and $\delta_t > 1$ for $t > 0$, i.e., the *hypotheses* such that one or more theoretical peaks are not accessed, where a hypothesis is an assignment of all random variables in the graph. The number of deletions occurring in a single frame is then δ_0 for the prologue and $(\delta_t - 1)\mathbf{1}\{\delta_t > 1\}$ for all subsequent frames, where $\mathbf{1}\{\cdot\}$ is the indicator function which returns 1 if its argument is true and 0 otherwise. The total number of allowed deletions is \mathcal{Z} and counts down in subsequent frames such that, denoting the number of deletions left in a frame as Z_t , we have

$$p(Z_0 = \mathcal{Z} - \delta_0 | \mathcal{Z}, \delta_0) = 1$$

$$p(Z_t = Z_{t-1} - (\delta_t - 1)\mathbf{1}\{\delta_t > 1\} | Z_{t-1}, \delta_t) = 1, t > 0.$$

The allowable number of insertions counts down in a similar manner to the deletions. ξ_0 is the maximum allowable insertions for all frames, i_t is a Bernoulli random variable which signifies whether the peak in frame t is an insertion, and $p(\xi_t = \xi_{t-1} - i_{t-1} | \xi_{t-1}, i_{t-1}) = 1$. Furthermore, the role of ξ_t as a switching parent of i_t is such that $p(i_t = 0 | \xi_t = 0) = 1$. Thus, when there are no insertions left, i_t is 0 for all remaining frames.

The hidden multinomial δ_t is such that $p(\delta_0 > \mathcal{Z}) = 0$, i.e. it respects the maximum deletion constraint of the first frame, and for $t > 0$, $p(\delta_t) = \sum_{i_{t-1}} p(\delta_t | \delta_{t-1}, Z_{t-1}, \tilde{n}^x, i_{t-1}) p(i_{t-1})$, where

$$p(\delta_t = 0 | \delta_{t-1}, Z_{t-1}, \tilde{n}^x, i_{t-1} = 1) = 1, \quad (5)$$

$$p(\delta_t > \tilde{n}^x - (K_t - Z_t) | \tilde{n}^x, K_t, Z_t, i_{t-1}) = 0. \quad (6)$$

Equation (5) prohibits DRIP from moving down the theoretical spectrum in a frame following an insertion. This constraint ensures that the theoretical spectrum may not be trivially traversed while observed peaks are scored as insertions, or equivalently that some observed peak must not be scored as an insertion in order to move down the theoretical spectrum for frames $t > 0$. Equation (6) constrains DRIP from incrementing past the range of valid theoretical peak indices.

Table 1: Random variable hypothesis for alignment displayed in Figure 5. Recall the deterministic relationships $K_0 = \delta_0$, $K_t = K_{t-1} + \delta_t$ for $t > 0$, and given the theoretical peak index K_t we have the theoretical peak $v^x(K_t)$. For instance, from the theoretical spectrum of $x = \text{EALK}$ in Figure 4, we have $K_6 = K_5 + \delta_6$ and $v^x(K_6) = v^x(1) = 147$.

t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
δ_t	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	2	0	0	0	0	0	1	0	0	0	0	0
K_t	0	0	0	0	0	0	1	1	1	2	2	2	2	2	2	4	4	4	4	4	4	4	5	5	5	5	5
i_t	1	1	1	1	1	0	1	0	0	1	1	1	1	0	0	1	1	1	1	1	1	0	1	1	0	1	1

In order to discourage the use of deletions unless absolutely necessary, the distribution over δ_t is constrained to be monotone decreasing, such that for $i_{t-1} = 1$, $K_t - Z_t < \tilde{n}^x$, and $0 < h < \tilde{n}_t^x - (K_t - Z_t) - 1$, we have $p(\delta_t = h | \tilde{n}^x, K_t, Z_t, i_{t-1}) < p(\delta_t = h - 1 | \tilde{n}^x, K_t, Z_t, i_{t-1})$.

The epilogue variable $R_{\tilde{n}^s}$, observed to 1, constrains which theoretical peaks may occur in the final frame. If the number of theoretical peaks left unexplored in the epilogue is greater than the number of remaining deletions, then such a hypothesis of random variables receives probability zero. Thus, we have $p(R_{\tilde{n}^s} = 1 | \tilde{n}^x, K_{\tilde{n}^s}, Z_{\tilde{n}^s}) = \mathbf{1}\{\tilde{n}^x - K_{\tilde{n}^s} \leq Z_{\tilde{n}^s}\}$. This boundary condition limits the number of valid hypotheses in DRIP which have non-zero probability, and by forcing the traversal of the theoretical spectrum from prologue to epilogue ensures that a peptide cannot align trivially well to the observed spectrum. Figure 4 and Table 1 detail the theoretical spectrum traversal for the alignment depicted in Figure 5.

4.2 SCORING OBSERVED PEAKS

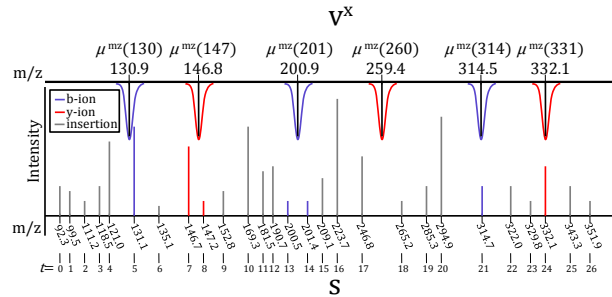


Figure 5: Spectra alignment in DRIP given observed spectrum s , $x = \text{EALK}$, the theoretical Gaussian peaks of v^x , and hypothesis from Figure 4. Note that t is the frame number, all Gaussians have equal variance, and $v^x(3)$ is a deletion.

The m/z observation is scored by a Gaussian centered near the theoretical peak accessed in a frame, and the intensity observation is scored using a Gaussian with mean greater than or equal to the most intense peak value. In this manner, DRIP aligns the theoretical and observed spectra by penalizing observed peaks far from theoretical peaks and penalizing peaks with intensity less than unity. The conditional

distributions of the observations are then

$$p(O_t^{\text{mz}} | v^x(K_t)) = \sum_{i_t} p(O_t^{\text{mz}} | v^x(K_t), i_t) p(i_t),$$

$$p(O_t^{\text{in}}) = \sum_{i_t} p(O_t^{\text{in}} | i_t) p(i_t).$$

When $i_t = 0$, i.e., the t th observed peak is not an insertion, then the observations are scored as

$$p(O_t^{\text{mz}} | v^x(K_t), i_t = 0) \sim \mathcal{N}(\mu^{\text{mz}}(v^x(K_t)), \sigma^2),$$

$$p(O_t^{\text{in}} | \mu^{\text{in}}, i_t = 1) \sim \mathcal{N}(\mu^{\text{in}}, \bar{\sigma}^2),$$

where μ^{in} and $\bar{\sigma}^2$ are the mean and variance of the Gaussian used to score peak intensities, σ^2 is the variance of the Gaussian used to score m/z observations, and μ^{mz} is a vector of means such that an arbitrary theoretical peak j scores m/z observations using $\mathcal{N}(\mu^{\text{mz}}(j), \sigma^2)$. The theoretical peaks serve as indices into a vector of Gaussians, each such Gaussian having equal variance and centered at a unique m/z position (illustrated in Figure 5). Thus, when describing a theoretical peak as scoring an observed peak, we are referring to scoring using the Gaussian accessed by the theoretical peak. To avoid confusion, we refer to the Gaussian accessed by a theoretical peak as the *theoretical Gaussian peak*.

All DRIP Gaussian means and variances are learned using expectation-maximization (EM) [14] and a high-confidence set of PSMs used in [9]. Note that this adds a great deal of modeling power to DRIP, allowing for adaptability of the expected m/z location of theoretical peaks. In MS/MS data, the m/z measurements will be stochastically offset from the true m/z values in a manner dependent upon machine precision, and may also be systematically offset in a non-linear fashion due to machine miscalibration [15]. Learning the means of the Gaussians allows DRIP to account for and adapt to these trends. Furthermore, the learned means themselves may be of interest to researchers in the field as a method of studying the nonlinear warping of the m/z axis encountered in specific experiments.

4.2.1 Insertion penalties

Due to the noisy nature of MS/MS data, only a small minority of observed peaks actually correspond to fragment ions. As an example, Figure 1 displays excellent fragmentation of the peptide present in the observed spectrum. However,

only 22 of 324 peaks correspond to fragment ions. Furthermore, these fragment ions only occur within a sufficiently small Th window of their theoretical values. Indeed, the learned variance σ^2 dictates that 99.9937% of the mass for a theoretical Gaussian peak lies within an approximately 1Th range, so that attempting to score all m/z observations with theoretical Gaussian peaks would see a majority of peptides score poorly for almost all observed spectra. Such an approach would also make the comparison of scores across different spectra in the same dataset difficult, because the variance among the PSMs would be incredibly high. Thus, when $i_t = 1$, the observations are scored as

$$\begin{aligned} p(O_t^{\text{mz}}|v^x(K_t), i_t = 1) &= p(O_t^{\text{mz}}|i_t = 1) = a \\ p(O_t^{\text{in}}|i_t = 1) &= b, \end{aligned}$$

where a and b are constants.

The insertion constant a imposes a tradeoff between, on the one hand, receiving exponentially bad scores from scoring observed peaks far from theoretical Gaussian peaks and, on the other hand, simply receiving an arbitrarily large constant penalty. To balance this tradeoff, a is set to the score received evaluating an m/z observation 4 standard deviations from the theoretical Gaussian peak mean, $\mu^{\text{mz}}(v^x(K_t))$, i.e., $a = f(4\sigma - \mu^{\text{mz}}(v^x(K_t))|\mu^{\text{mz}}(v^x(K_t)), \sigma^2)$, where $f(z|\mu, \sigma^2)$ is the scalar Gaussian with mean μ and variance σ^2 , evaluated at $z \in \mathbb{R}$. Thus, scoring an m/z observation score is greater than a so long as the observation remains within 99.9937% of the centered mass of $\mathcal{N}(\mu^{\text{mz}}(v^x(K_t)), \sigma^2)$. Similarly, the penalty b is set such that an intensity observation score is greater than b so long as it is within a specified percentage of the centered mass of $\mathcal{N}(\mu^{\text{in}}, \bar{\sigma}^2)$. The percentage used for the results in Section 5 was 20%, prioritizing aligning the observed and theoretical peaks over simply scoring high intensity peaks. Furthermore, the number of allowable insertions is limited per peptide (described in [10]), restricting the ability of arbitrary peptides to score observed peaks well.

4.3 DRIP SCORING FUNCTION

Peptides are scored by their optimal alignment using their per-frame log-Viterbi Score,

$$\psi(s, x) = \frac{1}{n^s} \max_{\delta_t, i_t, \forall t} \log p(s|x) = \frac{1}{n^s} \log p^*(s|x). \quad (7)$$

Dividing by the number of frames allows comparability of PSMs from different spectra. In order to further analyze the scoring function, assume inference has been completed and we have computed the Viterbi path, using $*$ to denote a variable's Viterbi value. Let $\lambda = \sum_{t=0}^{\tilde{n}^s} i_t^*$ denote the number of used insertions and note that $p(i_t = 0) = p(i_0 =$

0). DRIP's score is then

$$\begin{aligned} \log p^*(s|x) &= \lambda[\log(ab) + 3 \log p(i_0 = 1)] + 3(n^s - \lambda) \log p(i_0 = 0) + \\ &\sum_{t=0}^{\tilde{n}^s} \left[\log p(\delta_t^*) + \log f(O_t^{\text{in}}|\mu^{\text{in}}, \sigma^2) + \right. \\ &\quad \left. \mathbf{1}\{i_t^* = 0\}(\log f(O_t^{\text{mz}}|\mu^{\text{mz}}(v^x(K_t^*)), \sigma^2)) \right] \end{aligned} \quad (8)$$

where, as before, $f(z|\mu, \sigma^2)$ is the scalar Gaussian with mean μ and variance σ^2 , evaluated at $z \in \mathbb{R}$. The learned model variances are such that $\sigma^2 < \bar{\sigma}^2$, i.e., there is more uncertainty in intensity measurements than m/z measurements. Thus, it is easy to see that when a peptide does not align well with the observed spectrum (i.e., many observed peaks are far from the closest theoretical peak), then the $\log f(O_t^{\text{mz}}|\mu^{\text{mz}}(v^x(K_t^*)), \sigma^2)$ term severely penalizes the score. Furthermore, this score decreases quickly as the distance between the observed peak and theoretical peak mean increases. This also implies that peptides which arbitrarily match intense peaks will still receive poor scores if they do not align well.

4.4 APPROXIMATE INFERENCE

Table 2: DRIP per-peptide run-times (in seconds) for 3 yeast spectra, 1000 scored candidate peptides each.

Spec.	Exact Inf.	$k = 1500$	$k = 1000$	$k = 500$
s_1	0.07816	0.01056	0.00779	0.00436
s_2	0.30003	0.02070	0.01496	0.00820
s_3	3.61777	0.04105	0.02861	0.01586

The state space of the random variables in DRIP grows rapidly as the number of observed and theoretical peaks increases. Although the observed variables $\tilde{n}^x, R_{\tilde{n}^x}, \mathcal{Z}, \xi_0$ greatly decrease the number of states necessary to explore by limiting the hypotheses in DRIP which receive non-zero probability, there are still an exponentially large number of states to score in order to find the Viterbi path. However, the problem of interest is ideally suited for approximate inference techniques, specifically beam pruning [16]. In beam pruning, assuming a beam width of $k \in \mathbb{N}$, only the top k most probable states in a frame are allowed to persist. Although under this methodology we are no longer theoretically guaranteed to find the Viterbi path, the structure of the problem and the value of the learned theoretical Gaussian variances ensures that, per frame, many of the hypotheses will be of extremely low probability.

For instance, the hypothesis that the first theoretical peak matches the last observed peak is highly improbable. In general, the hypothesis that a theoretical peak centered many Thomsons away from an observed peak is also highly improbable. Thus, we can retain the k most probable states in a frame without deleteriously affecting the Viterbi score.

However, care must be taken such that k is not too small or else globally good alignments where a frame must be explained by a low probability event may not be allowed to persist. Table 2 displays the per-peptide run times for 3 randomly chosen yeast spectra, scoring 1000 candidate peptides per spectrum using exact inference and various k values. Each test was performed on the same machine with an Intel Core 2 Quad Q9550 and 8GB RAM. For $k \in \{1500, 1000\}$, all peptide scores were equal to their exact scores. For $k = 500$, 0.1% of the peptide scores differed from their exact scores. The top ranking PSM scores did not change for all beam widths. The results found in Section 5 were generated using $k = 1500$.

4.5 DRIP OBSERVED SPECTRUM PREPROCESSING

As with all other search algorithms, we score database peptides of at most a fixed maximum length, specified prior to run time. Practical values of the maximum peptide length (the maximum peptide length considered for all results in Section 5 is 50) mean that the number of observed peaks is typically an order of magnitude larger than the number of theoretical peaks for any scored peptide. Furthermore, most of these peaks are noise peaks [17], and as such we filter all but the most l intense peaks, where in practice, $l = 300$.

After filtering peaks, the observed spectrum is renormalized as in SEQUEST [8], the steps of which are as follows. Firstly, all observed peak intensity values are replaced with their square root. Secondly, the observed spectrum is partitioned into 10 equally spaced regions along the m/z axis and all peaks in a region are normalized to some globally maximum intensity, which in our case is 1. Steps 1 and 2 greatly decrease the high variance of observed intensities, and step 2 helps ensure that scoring is not severely biased by many large peaks lying arbitrarily close to one another. Lastly, any peak with intensity less than $1/20$ of the most intense peak is filtered. Note that through all of these preprocessing steps, the m/z values for all remaining observed peaks remain unaltered and, as discussed in Section 4.3, the scoring of these unaltered values dominates the returned DRIP score.

5 RESULTS

We compared the performance of DRIP to four competing database search methods (Section 3). In these evaluations, we do not have an independently labeled gold standard set of identified spectra. Although it is possible to send a purified sample of known peptides through the MS/MS pipeline to obtain high confidence identifications, the low complexity of the input sample yields spectra that are less noisy than real spectra. Therefore, as is common in this field, we estimate for each search procedure the *false discovery rate* (i.e., the proportion of spectra above a given threshold that are incorrectly identified, or $1 - \text{precision}$) by searching a decoy

database of peptides [18]. These decoys are generated by shuffling the peptides in the target database. Because FDR is not monotonically related to the underlying score, we compute a q -value for each scored spectrum, defined as the minimum FDR threshold at which that score is deemed significant. Once the target and decoy PSMs are calculated, we plot the number of identified targets as a function of q -value threshold. In practice, search results with an FDR $> 10\%$ are not practically useful, so we only plot $q \in [0, 0.1]$.

We use eight previously described datasets [1] as well as another yeast dataset (yeast-03) taken from the same repository, considering only spectra with charge $2+$. All benchmark methods were searched using the same target and decoy databases, and all parameters across search algorithms were set as equivalently as possible. All datasets and reported PSMs per benchmark method may be found at <http://noble.gs.washington.edu/proj/drip>. As seen in the results panel in Figure 5, DRIP outperforms SEQUEST and OMSSA at all q -value thresholds on all datasets. DRIP is the most frequent top performer, beating all other methods on four datasets, compared to MS-GFDB and Didea, each of which is top performer on only two datasets. Furthermore, DRIP individually outperforms MS-GFDB and Didea on six of the nine datasets. DRIP also offers the most consistent performance compared to MS-GFDB and Didea across the different organisms: DRIP is always ranked first or second, whereas MS-GFDB and Didea rank third on many datasets. Both DRIP and Didea only model b- and y-ions, whereas the other algorithms [8, 11, 12] use more complicated models of peptide fragmentation (further discussed in [10]).

5.1 INSERTION, DELETION COUNT-BASED SCORES

Once a peptide’s Viterbi path has been decoded, the total number of insertions and deletions used by a peptide to score an observed spectrum may be calculated. These two quantities may be used as quality measures for a PSM, as well as to exactly compute the signal-to-noise ratio per observed spectrum. To illustrate the utility of these two quantities, we show that using both as scoring functions allows some discriminative power to differentiate between target and decoy peptides, outperforming OMSSA and SEQUEST as well as MS-GFDB over some datasets. Plotted in Figure 6, DRIP-NotDel and DRIP-NotIns correspond to scoring functions utilizing the number of a peptide’s theoretical peaks not deleted and the number of observed peaks a peptide did not consider an insertion, respectively. Note the piece-wise linear behavior, which is caused by scoring ties due to the scoring functions being integer based.

It is worth noting that scoring methods, such as SEQUEST and Didea, which perform binning typically do so by taking the maximum observed peak intensity falling within a bin. Under such binning schemes, the number of theoretical peaks not deleted is equal to the number of observed peaks which are not insertions. In DRIP, where quantization is

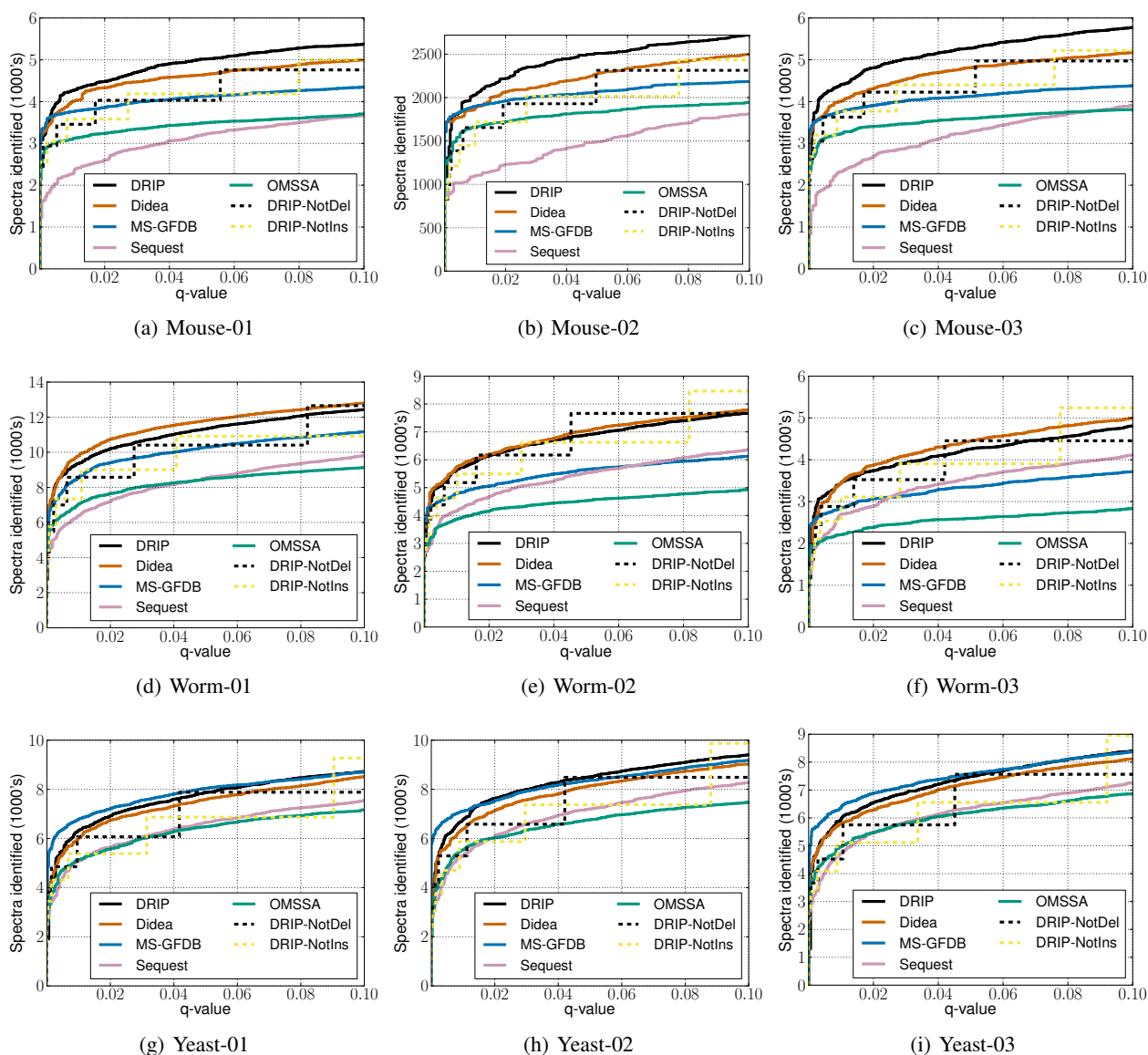


Figure 6: Performance curves for DRIP. The x-axis may be thought of as the significance threshold, and the y-axis the number of correctly identified spectra at a threshold. Thus, higher on the y-axis denotes better performance. DRIP-NotDel and DRIP-NotIns utilize the decoded DRIP Viterbi path to calculate the number of theoretical peaks not deleted and number of observed peaks not inserted, respectively, as scoring functions.

not performed, these two quantities are not equal (Figure 6). Such quantities are typically used by post-processors as features for the task of reranking target and decoy scores for improved accuracy [19, 20], and these quantities (and potentially others) calculated from DRIP’s Viterbi path may similarly be used as features.

5.2 IMPACT OF LEARNED PARAMETERS ON PERFORMANCE

The use of Gaussians allows DRIP to avoid quantization of m/z measurements, unlike all existing competitors. Learning the Gaussian means and variances in DRIP provides both a tool to study the nonlinear m/z offsets caused by machine error [15] as well as a significant increase in performance. As previously mentioned, the Gaussian parameters are learned using EM and a high-confidence set of charge 2 PSMs used in [9]. Figure 7 displays the performance benefits of jointly learning these parameters.

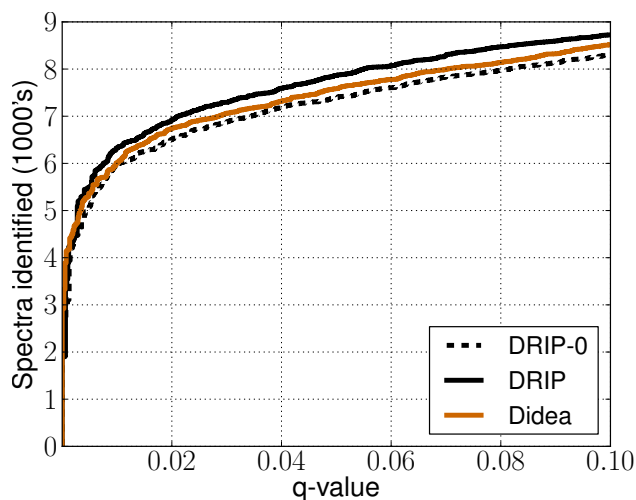


Figure 7: DRIP run with different model parameters over Yeast-01, where DRIP-0 consists of hand-tuned Gaussian parameters and DRIP consists of EM jointly learned Gaussian means and variances.

DRIP-0 consists of setting the Gaussian means to values halfway between integer units along the m/z axis and setting the intensity variance to an order of magnitude larger than the m/z variance, so as to penalize misalignment between the theoretical and observed spectra greater than small intensities. DRIP consists of jointly learning both the Gaussian means and variances, the parameters used for testing over all datasets in Figure 6. Interestingly, the learned intensity variance is larger than the learned m/z variance, so that the learned parameters dictate m/z measurements have the largest impact when scoring. Jointly learning both the means and variances improves performance compared to hand-tuned parameters, leading to improved performance relative to Didea. This trend of improved performance via learning the Gaussian parameters is observed on the other

datasets, as well.

6 CONCLUSIONS AND FUTURE WORK

We have presented DRIP, a generative model of peptide fragmentation in MS/MS. Through DRIP, a database peptide is scored by maximally aligning the peptide’s theoretical spectrum to the observed MS/MS spectrum via Viterbi decoding. Unlike previous database search methods, the observed spectrum is not quantized; instead, the m/z measurements are scored in their natural resolution. Considering the recent push in the field toward high-resolution data [21], for which other search methods must reevaluate their quantization schemes, DRIP’s handling of m/z values at full resolution is particularly important.

DRIP’s scoring function outperforms state-of-the-art algorithms on many of the presented datasets, and is far superior to the popular search algorithms SEQUEST and OMSSA. Furthermore, unlike a recent DBN-based database search method [1], DRIP is a highly trainable model which allows it a great deal of adaptability to the wide variety of machines and experimental conditions. The Viterbi path calculated in DRIP also provides a large amount of information, which otherwise typically requires post-processing after database search. Finally, via sum-product inference, DRIP may be used to calculate posteriors of particular interest to end users, a task which has previously required complicated post-processing [5].

We plan to pursue several avenues for future work. Initially, we will collect high-quality training sets of PSMs charge states other than 2 and for high-resolution spectra. Perhaps the most exciting avenue for future work is that a minor change to the DRIP model will allow it to align an observed spectrum to not just one but many different peptides simultaneously. We plan to investigate sequential variants of algebraic decision diagrams [22] to represent (potentially exponentially) large collections of peptides in polynomial space and to exploit the dynamic programming nature of DBNs to be able to score such peptide collections efficiently. Such a framework will also generalize to de novo sequencing, in which we search over the set of all possible peptides as opposed to simply a database. Finally, we plan to investigate generalizing the use of algebraic decision diagrams to allow DRIP to calibrate its scores relative to the entire peptide set. This would be similar in spirit to the dynamic programming calibration employed by methods like MS-GFDB.

Acknowledgements: This work was supported by the National Institutes of Health (NIH) under awards R01 GM096306, P41 GM103533, and T32 HG00035, and by the National Science Foundation (NSF) under grant CNS-0855230.

References

- [1] A. P. Singh, J. Halloran, J. A. Bilmes, K. Kirchoff, and W. S. Noble, "Spectrum identification using a dynamic bayesian network model of tandem mass spectra," in *Uncertainty in Artificial Intelligence (UAI)*, (Catalina Island, USA), AUAI, July 2012.
- [2] K. Filali and J. Bilmes, "A Dynamic Bayesian Framework to Model Context and Memory in Edit Distance Learning: An Application to Pronunciation Classification," in *Proceedings of the Association for Computational Linguistics (ACL)*, 43, (University of Michigan, Ann Arbor), 2005.
- [3] J. Pearl, *Probabilistic Reasoning in Intelligent Systems : Networks of Plausible Inference*. Morgan Kaufmann, 1988.
- [4] J. Bilmes, "Dynamic graphical models," *IEEE Signal Processing Magazine*, vol. 27, pp. 29–42, Nov 2010.
- [5] S. A. Beausoleil, J. Villen, S. A. Gerber, J. Rush, and S. P. Gygi, "A probability-based approach for high-throughput protein phosphorylation analysis and site localization," *Nature Biotechnology*, vol. 24, no. 10, pp. 1285–1292, 2006.
- [6] H. Steen and M. Mann, "The ABC's (and XYZ's) of peptide sequencing," *Nature Reviews Molecular Cell Biology*, vol. 5, pp. 699–711, 2004.
- [7] M. Kinter and N. E. Sherman, *Protein sequencing and identification using tandem mass spectrometry*. Wiley-Interscience, 2000.
- [8] J. K. Eng, A. L. McCormack, and J. R. Yates, III, "An approach to correlate tandem mass spectral data of peptides with amino acid sequences in a protein database," *Journal of the American Society for Mass Spectrometry*, vol. 5, pp. 976–989, 1994.
- [9] A. A. Klammer, S. Reynolds, M. J. MacCoss, J. Bilmes, and W. S. Noble, "Improved peptide identification and spectrum prediction using a probabilistic model of peptide fragmentation," in *ASMS*, 2006.
- [10] J. T. Halloran, J. A. Bilmes, and W. S. Noble, "Learning Peptide-Spectrum Alignment Models for Tandem Mass Spectrometry: Extended Version," 2014.
- [11] L. Y. Geer, S. P. Markey, J. A. Kowalak, L. Wagner, M. Xu, D. M. Maynard, X. Yang, W. Shi, and S. H. Bryant, "Open mass spectrometry search algorithm," *Journal of Proteome Research*, vol. 3, pp. 958–964, 2004. OMSSA.
- [12] S. Kim, N. Mischerikow, N. Bandeira, J. D. Navarro, L. Wich, S. Mohammed, A. J. R. Heck, and P. A. Pevzner, "The generating function of cid, etd, and cid/etd pairs of tandem mass spectra: Applications to database search," *Molecular and Cellular Proteomics*, vol. 9, no. 12, pp. 2840–2852, 2010.
- [13] J. Bilmes, "Dynamic Bayesian Multinets," in *UAI '00: Proceedings of the 16th Conference in Uncertainty in Artificial Intelligence* (C. Boutilier and M. Goldszmidt, eds.), (San Francisco, CA, USA), Morgan Kaufmann Publishers, 2000.
- [14] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, pp. 1–22, 1977.
- [15] J. D. Egerton, J. K. Eng, M. S. Bereman, E. J. Hsieh, G. E. Merrihew, and M. J. MacCoss, "De novo correction of mass measurement error in low resolution tandem ms spectra for shotgun proteomics," *Journal of The American Society for Mass Spectrometry*, pp. 1–8, 2012.
- [16] H. Ney and S. Ortmanns, "Progress in dynamic programming search for lvsr," *Proceedings of the IEEE*, vol. 88, no. 8, pp. 1224–1240, 2000.
- [17] B. Y. Renard, M. Kirchner, F. Monigatti, A. R. Ivanov, J. Rappsilber, D. Winterc, J. A. Steen, F. A. Hamprecht, and H. Steen, "When less can yield more—computational preprocessing of ms/ms spectra for peptide identification," *Proteomics*, vol. 9, no. 21, pp. 4978–4984, 2009.
- [18] L. Käll, J. D. Storey, M. J. MacCoss, and W. S. Noble, "Assigning significance to peptides identified by tandem mass spectrometry using decoy databases," *Journal of Proteome Research*, vol. 7, no. 1, pp. 29–34, 2008.
- [19] L. Käll, J. Canterbury, J. Weston, W. S. Noble, and M. J. MacCoss, "A semi-supervised machine learning technique for peptide identification from shotgun proteomics datasets," *Nature Methods*, vol. 4, pp. 923–25, 2007.
- [20] A. Keller, A. I. Nesvizhskii, E. Kolker, and R. Aebersold, "Empirical statistical model to estimate the accuracy of peptide identification made by MS/MS and database search," *Analytical Chemistry*, vol. 74, pp. 5383–5392, 2002.
- [21] C. D. Wenger and J. J. Coon, "A proteomics search algorithm specifically designed for high-resolution tandem mass spectra," *Journal of proteome research*, 2013.
- [22] R. I. Bahar, E. A. Frohm, C. M. Gaona, G. D. Hachtel, E. Macii, A. Pardo, and F. Somenzi, "Algebraic decision diagrams and their applications," in *Computer-Aided Design, 1993. ICCAD-93. Digest of Technical Papers., 1993 IEEE/ACM International Conference on*, pp. 188–191, IEEE, 1993.

Off-policy TD(λ) with a true online equivalence

Hado van Hasselt

A. Rupam Mahmood

Richard S. Sutton

Reinforcement Learning and Artificial Intelligence Laboratory
University of Alberta, Edmonton, AB T6G 2E8 Canada

Abstract

Van Seijen and Sutton (2014) recently proposed a new version of the linear TD(λ) learning algorithm that is exactly equivalent to an online forward view and that empirically performed better than its classical counterpart in both prediction and control problems. However, their algorithm is restricted to on-policy learning. In the more general case of off-policy learning, in which the policy whose outcome is predicted and the policy used to generate data may be different, their algorithm cannot be applied. One reason for this is that the algorithm bootstraps and thus is subject to instability problems when function approximation is used. A second reason true online TD(λ) cannot be used for off-policy learning is that the off-policy case requires sophisticated importance sampling in its eligibility traces. To address these limitations, we generalize their equivalence result and use this generalization to construct the first online algorithm to be exactly equivalent to an off-policy forward view. We show this algorithm, named *true online GTD(λ)*, empirically outperforms GTD(λ) (Maei, 2011) which was derived from the same objective as our forward view but lacks the exact online equivalence. In the general theorem that allows us to derive this new algorithm, we encounter a new general eligibility-trace update.

1 Temporal difference learning

Eligibility traces improve learning in temporal-difference (TD) algorithms by efficiently propagating credit for later observations back to update earlier predictions (Sutton, 1988), and can help speed up learning significantly. A good way to interpret these traces, the extent of which is regulated by a trace parameter $\lambda \in [0, 1]$, is to consider the eventual updates to each prediction. For $\lambda = 1$ the up-

date for the prediction at time t is similar to a Monte Carlo update towards the full return following t . For $\lambda = 0$ the prediction is updated toward only the immediately (reward) signal, and the rest of the return is estimated with the prediction at the next state. Such an interpretation is called a forward view, because it considers the effect of future observations on the updates. In practice, learning is often fastest for intermediate values of λ (Sutton & Barto, 1998).

Traditionally, the equivalence to a forward view was known to hold only when the predictions are updated offline. In practice TD algorithms are more commonly used online, during learning, but then this equivalence was only approximate. Recently, van Seijen and Sutton (2014) developed true online TD(λ), the first algorithm to be exactly equivalent to a forward view under online updating. For $\lambda = 1$ the updates by true online TD(λ) eventually become exactly equivalent to a Monte Carlo update towards the full return. As demonstrated by van Seijen and Sutton, such an online equivalence is more than a theoretical curiosity, and leads to lower prediction errors than when using the traditional TD(λ) algorithm that only achieves an offline equivalence. In this paper, we generalize this result and show exact online equivalences are possible for a wide range of forward views, leading to computationally efficient online algorithms by exploiting a new generic trace update.

A limitation of the true online TD(λ) algorithm by van Seijen and Sutton (2014) is that it is only applicable to on-policy learning, when the learned predictions correspond to the policy that is used to generate the data. Off-policy learning is important to be able to learn from demonstrations, to learn about many things at the same time (Sutton et al., 2011), and ultimately to learn about the unknown optimal policy. A natural next step is therefore to apply our general equivalence result to an off-policy forward view. We construct such a forward view and derive an equivalent new off-policy gradient TD algorithm, that we call *true online GTD(λ)*. This algorithm is constructed to be equivalent for $\lambda = 0$, by design, to the existing GTD(λ) algorithm (Maei, 2011). We demonstrate empirically that for higher λ the new algorithm is much better behaved due to its exact

equivalence to a desired forward view. In addition to the practical potential of the new algorithm, this demonstrates the usefulness of our general equivalence result and the resulting new trace update.

2 Problem setting

We consider a learning agent in an unknown environment where at each time step t the agent performs an action A_t after which the environment transitions from the current state S_t to the next state S_{t+1} . We do not assume the state itself can be observed and the agent instead observes a feature vector $\phi_t \in \mathbb{R}^n$, which is typically a function of the state S_t such that $\phi_t \doteq \phi(S_t)$. The agent selects its actions according to a behavior policy b , such that $b(a|S_t)$ denotes the probability of selecting action $A_t = a$ in state S_t . Typically $b(a|s)$ depends on s through $\phi(s)$.

After performing A_t , the agent observes a scalar (reward) signal R_{t+1} and the process can either terminate or continue. We allow for soft terminations, defined by a potentially time-varying state-dependent termination factor $\gamma_t \in [0, 1]$ (cf. Sutton, Mahmood, Precup & van Hasselt, 2014). With weight $1 - \gamma_{t+1}$ the process terminates at time $t + 1$ and R_{t+1} is considered the last reward in this episode. With weight γ_{t+1} we continue to the next state and observe $\phi_{t+1} \doteq \phi(S_{t+1})$. The agent then selects a new action A_{t+1} and this process repeats. A special case is the episodic setting where $\gamma_t = 1$ for all non-terminating times t and $\gamma_T = 0$ when the episode ends at time T . The termination factors are commonly called *discount factors*, because they discount the effect of later rewards.

The goal is to predict the sum of future rewards, discounted by the probabilities of termination, under a target policy π . The optimal prediction is thus defined for each state s by

$$v_\pi(s) \doteq \mathbb{E}_\pi \left[\sum_{t=1}^{\infty} R_t \prod_{k=1}^t \gamma_k \mid S_0 = s \right],$$

where $\mathbb{E}_\pi[\cdot] \doteq \mathbb{E}[\cdot \mid A_t \sim \pi(\cdot|S_t), \forall t]$ is the expectancy conditional on the policy π . We estimate the values $v_\pi(s)$ with a parameterized function of the observed features. In particular we consider linear functions of the features, such that $\theta_t^\top \phi_t \approx v_\pi(S_t)$ is the estimated value of the state at time t according to a weight vector θ_t . The goal is then to improve the predictions by updating θ_t .

We desire online algorithms with a constant $O(n)$ per-step complexity, where n is the number of features in ϕ . Such computational considerations are important in settings with a lot of data or when ϕ_t is a large vector. For instance, we want our algorithms to be able to run on a robot with many sensors and limited on-board processing power.

3 General online equivalence between forward and backward views

We can think about what the ideal update would be for a prediction after observing all relevant future rewards and states. Such an update is called a forward view, because it depends on observations from the future. A concrete example is the on-policy Monte Carlo return, consisting of the discounted sum of all future rewards.

In practice, full Monte Carlo updates can have high variance. It can be better to augment the return with the then-current predictions at the visited states. When we continue after some time step t , with weight γ_{t+1} , we replace a portion of $1 - \lambda_{t+1}$ of the remaining return with our current prediction of this return at S_{t+1} . Making use of later predictions to update earlier predictions in this way is called *bootstrapping*. The process then continues to the next action and reward with total weight $\gamma_{t+1}\lambda_{t+1}$, where again we terminate with $1 - \gamma_{t+2}$ and then bootstrap with $1 - \lambda_{t+2}$, and so on. When $\lambda_{t+1} = 0$ we get the usual one-step TD return $R_{t+1} + \gamma_{t+1}\phi_{t+1}^\top \theta_t$. If $\lambda_t = 1$ for all t , we obtain a full (discounted) Monte Carlo return. In the on-policy setting, when we do not have to worry about deviations from the target policy, we can then update the prediction made at time t towards the on-policy λ -return defined by

$$G_t^\lambda = R_{t+1} + \gamma_{t+1} [(1 - \lambda_{t+1})\phi_{t+1}^\top \theta_t + \lambda_{t+1}G_{t+1}^\lambda].$$

The discount factors γ_t are normally considered a property of the problem, but the bootstrap parameters λ_t can be considered tunable parameters. The full return (obtained for $\lambda = 1$) is an unbiased estimate for the value of the behavior policy, but its variance can be high. The value estimates are typically not unbiased, but can be considerably less variable. As such, one can interpret the λ parameters as trading off bias and variance. Typically, learning is fastest for intermediate values of λ .

If termination never occurs, G_t^λ is never fully defined. To construct a well-defined forward view, we can truncate the recursion at the current data horizon (van Seijen & Sutton, 2014; Sutton et al., 2014) to obtain interim λ -returns. If we have data up to time t , all returns are truncated as if $\lambda_t = 0$ and we bootstrap on the most recent value estimate $\phi_t^\top \theta_{t-1}$ of the current state. This gives us, for each $0 \leq k < t$

$$G_{k,t}^\lambda = R_{k+1} + \gamma_{k+1} [(1 - \lambda_{k+1})\phi_{k+1}^\top \theta_k + \lambda_{k+1}G_{k+1,t}^\lambda]$$

and $G_{t,t}^\lambda = \phi_t^\top \theta_{t-1}$. In this definition of $G_{k,t}^\lambda$, for each time step j with $k < j \leq t$ the value of state S_j is estimated using $\phi_j^\top \theta_{j-1}$, because θ_{j-1} is the most up-to-date weight vector at the moment we reach this state.

Using these interim returns, we can construct an interim forward view which, in contrast to conventional forward

views, can be computed before an episode has concluded or even if the episode never fully terminates. For instance, when we have data up to time t the following set of linear updates for all times $k < t$ is an interim forward view:

$$\theta_{k+1}^t = \theta_k^t + \alpha_k (G_{k,t}^\lambda - \phi_k^\top \theta_k^t) \phi_k, \quad k < t, \quad (1)$$

where $\theta_0^t \doteq \theta_0$ is the initial weight vector. The subscript on θ_k^t (first index on $G_{k,t}^\lambda$) corresponds to the state for the k th update, the superscript (second index on $G_{k,t}^\lambda$) denotes the current data horizon.

The forward view (1) is well-defined and computable at every time t , but it is not very computationally efficient. For each new observation, when t increments to $t+1$, we potentially have to recompute all the updates, as $G_{k,t+1}^\lambda$ might differ from $G_{k,t}^\lambda$ for arbitrary many k . The resulting computational complexity is $O(nt)$ per time step, which is problematic when t becomes large. Therefore, forward views are not meant to be implemented as is. They serve as a conceptual update, in which we formulate what we want to achieve after observing the relevant data.

In the next theorem, we prove that for many forward views an efficient and fully equivalent *backward view* exists that exploits eligibility traces to construct online updates that use only $O(n)$ computation per time step, but that still result in exactly the same weight vectors. The theorem is constructive, allowing us to find such backward views automatically for a given forward view.

Theorem 1 (Equivalence between forward and backward views). *Consider any forward view that updates towards some interim targets Y_k^t with*

$$\theta_{k+1}^t = \theta_k^t + \eta_k (Y_k^t - \phi_k^\top \theta_k^t) \phi_k + \mathbf{x}_k, \quad 0 \leq k < t,$$

where $\theta_0^t = \theta_0$ for some initial θ_0 and where $\mathbf{x}_k \in \mathbb{R}^n$ is any vector that does not depend on t . Assume that the temporal differences $Y_k^{t+1} - Y_k^t$ for different k are related through

$$Y_k^{t+1} - Y_k^t = c_k (Y_{k+1}^{t+1} - Y_{k+1}^t), \quad \forall k < t, \quad (2)$$

where c_k is a scalar that does not depend on t . Then, the final weights θ_t^t at each t are equal to the weights θ_t as defined by $\mathbf{e}_0 = \eta_0 \phi_0$ and the backward view

$$\begin{aligned} \theta_{t+1} &= \theta_t + (Y_t^{t+1} - Y_t^t) \mathbf{e}_t + \eta_t (Y_t^t - \phi_t^\top \theta_t) \phi_t + \mathbf{x}_t, \\ \mathbf{e}_t &= c_{t-1} \mathbf{e}_{t-1} + \eta_t (1 - c_{t-1} \phi_t^\top \mathbf{e}_{t-1}) \phi_t, \quad t > 0. \end{aligned} \quad (3)$$

Proof. We introduce the fading matrix $\mathbf{F}_t \doteq \mathbf{I} - \eta_t \phi_t \phi_t^\top$, such that $\theta_{k+1}^t = \mathbf{F}_k \theta_k^t + \eta_k Y_k^t \phi_k$. We subtract θ_t^t from θ_{t+1}^{t+1} to find the change when t increments. Expanding

θ_{t+1}^{t+1} , we get

$$\begin{aligned} \theta_{t+1}^{t+1} - \theta_t^t &= \mathbf{F}_t \theta_t^{t+1} - \theta_t^t + \eta_t Y_t^{t+1} \phi_t + \mathbf{x}_t \\ &= \mathbf{F}_t (\theta_t^{t+1} - \theta_t^t) + \eta_t Y_t^{t+1} \phi_t + (\mathbf{F}_t - \mathbf{I}) \theta_t^t + \mathbf{x}_t \\ &= \mathbf{F}_t (\theta_t^{t+1} - \theta_t^t) + \eta_t Y_t^{t+1} \phi_t - \eta_t \phi_t \phi_t^\top \theta_t^t + \mathbf{x}_t \\ &= \mathbf{F}_t (\theta_t^{t+1} - \theta_t^t) + \eta_t (Y_t^{t+1} - \phi_t^\top \theta_t^t) \phi_t + \mathbf{x}_t. \end{aligned} \quad (4)$$

We now repeatedly expand both θ_t^{t+1} and θ_t^t to get

$$\begin{aligned} \theta_t^{t+1} - \theta_t^t &= \mathbf{F}_{t-1} (\theta_{t-1}^{t+1} - \theta_{t-1}^t) + \eta_{t-1} (Y_{t-1}^{t+1} - Y_{t-1}^t) \phi_{t-1} \\ &= \mathbf{F}_{t-1} \mathbf{F}_{t-2} (\theta_{t-1}^{t+1} - \theta_{t-1}^t) \\ &\quad + \eta_{t-2} (Y_{t-2}^{t+1} - Y_{t-2}^t) \mathbf{F}_{t-1} \phi_{t-2} \\ &\quad + \eta_{t-1} (Y_{t-1}^{t+1} - Y_{t-1}^t) \phi_{t-1} \\ &= \dots \text{ (Expand until reaching } \theta_0^{t+1} - \theta_0^t = \mathbf{0}.) \\ &= \mathbf{F}_{t-1} \dots \mathbf{F}_0 (\theta_0^{t+1} - \theta_0^t) \\ &\quad + \sum_{k=0}^{t-1} \eta_k \mathbf{F}_{t-1} \dots \mathbf{F}_{k+1} (Y_k^{t+1} - Y_k^t) \phi_k \\ &= \sum_{k=0}^{t-1} \eta_k \mathbf{F}_{t-1} \dots \mathbf{F}_{k+1} (Y_k^{t+1} - Y_k^t) \phi_k \\ &= \sum_{k=0}^{t-1} \eta_k \mathbf{F}_{t-1} \dots \mathbf{F}_{k+1} c_k (Y_{k+1}^{t+1} - Y_{k+1}^t) \phi_k \quad \text{(Using (2))} \\ &= \dots \text{ (Apply (2) repeatedly.)} \\ &= c_{t-1} \sum_{k=0}^{t-1} \eta_k \underbrace{\left(\prod_{j=k}^{t-2} c_j \right)}_{\doteq \mathbf{e}_{t-1}} \mathbf{F}_{t-1} \dots \mathbf{F}_{k+1} \phi_k (Y_t^{t+1} - Y_t^t) \\ &= c_{t-1} \mathbf{e}_{t-1} (Y_t^{t+1} - Y_t^t). \end{aligned} \quad (5)$$

The vector \mathbf{e}_t can be computed with the recursion

$$\begin{aligned} \mathbf{e}_t &= \sum_{k=0}^t \eta_k \left(\prod_{j=k}^{t-1} c_j \right) \mathbf{F}_t \dots \mathbf{F}_{k+1} \phi_k \\ &= \sum_{k=0}^{t-1} \eta_k \left(\prod_{j=k}^{t-1} c_j \right) \mathbf{F}_t \dots \mathbf{F}_{k+1} \phi_k + \eta_t \phi_t \\ &= c_{t-1} \mathbf{F}_t \sum_{k=0}^{t-1} \eta_k \left(\prod_{j=k}^{t-2} c_j \right) \mathbf{F}_{t-1} \dots \mathbf{F}_{k+1} \phi_k + \eta_t \phi_t \\ &= c_{t-1} \mathbf{F}_t \mathbf{e}_{t-1} + \eta_t \phi_t \\ &= c_{t-1} \mathbf{e}_{t-1} + \eta_t (1 - c_{t-1} \phi_t^\top \mathbf{e}_{t-1}) \phi_t. \end{aligned}$$

We plug (5) back into (4) and obtain

$$\begin{aligned}
& \theta_{t+1}^{t+1} - \theta_t^t \\
&= c_{t-1} \mathbf{F}_t \mathbf{e}_{t-1} (Y_t^{t+1} - Y_t^t) + \eta_t (Y_t^{t+1} - \phi_t^\top \theta_t) \phi_t + \mathbf{x}_t \\
&= (\mathbf{e}_t - \eta_t \phi_t) (Y_t^{t+1} - Y_t^t) + \eta_t (Y_t^{t+1} - \phi_t^\top \theta_t) \phi_t + \mathbf{x}_t \\
&= (Y_t^{t+1} - Y_t^t) \mathbf{e}_t + \eta_t (Y_t^t - \phi_t^\top \theta_t) \phi_t + \mathbf{x}_t.
\end{aligned}$$

Because $\theta_{0,t} \doteq \theta_0$ for all t , the desired result follows through induction. \square

The theorem shows that under condition (2) we can turn a general forward view into an equivalent online algorithm that only uses $O(n)$ computation per time step. Compared to previous work on forward/backward equivalences, this grants us two important things. First, the obtained equivalence is both online and exact; most previous equivalences were only exact under offline updating, when the weights are not updated during learning (Sutton & Barto, 1998; Sutton et al., 2014). Second, the theorem is constructive, and gives an equivalent backward view directly from a desired forward view, rather than having to prove such an equivalence in hindsight (as in, e.g., van Seijen & Sutton, 2014). This is perhaps the main benefit of the theorem: rather than relying on insight and intuition to construct efficient online algorithms, Theorem 1 can be used to derive an exact backward view directly from a desired forward view. We exploit this in Section 6 when we turn a desired off-policy forward view into an efficient new online off-policy algorithm.

We refer to traces of the general form (3) as *dutch traces*. The trace update can be interpreted as first shrinking the traces with c , for instance $c = \gamma\lambda$, and then updating the traces for the current state, $\phi_t^\top \mathbf{e}$, towards one with a step size of η . In contrast, traditional accumulating traces, defined by $\mathbf{e}_t = c_{t-1} \mathbf{e}_{t-1} + \phi_t$, add to the trace value of the current state rather than updating it toward one. This can cause the accumulating traces to grow large, potentially resulting in high-variance updates.

To demonstrate one advantage of Theorem 1, we apply it to the on-policy TD(λ) forward view defined by (1).

Theorem 2 (Equivalence for true online TD(λ)). *Define $\theta_0^t = \theta_0$. Then, θ_t^t as defined by (1) equals θ_t as defined by the backward view*

$$\begin{aligned}
\delta_t &= R_{t+1} + \gamma \phi_{t+1}^\top \theta_t - \phi_t^\top \theta_{t-1}, \\
\mathbf{e}_t &= \gamma \lambda \mathbf{e}_{t-1} + \alpha_t (1 - \gamma \lambda \phi_t^\top \mathbf{e}_{t-1}) \phi_t, \\
\theta_{t+1} &= \theta_t + \delta_t \mathbf{e}_t + \alpha_t (\phi_t^\top \theta_{t-1} - \phi_t^\top \theta_t) \phi_t.
\end{aligned}$$

Proof. In Theorem 1, we substitute $\mathbf{x}_t = \mathbf{0}$, $c_t = \gamma\lambda$ and $Y_k^t = G_{k,t}^\lambda$, such that $Y_t^{t+1} - Y_t^t = \delta_t$ and $Y_t^t = \phi_t^\top \theta_{t-1}$. The desired result follows immediately. \square

The backward view in Theorem 2 is true online TD(λ), as proposed by van Seijen and Sutton (2014). Using Theorem 1, we have proved equivalence to its forward view with a few simple substitutions, whereas the original proof is much longer and more complex.

4 Off-policy learning

In this section, we turn to off-policy learning with function approximation. In constructing an off-policy forward view two issues arise that are not present in the on-policy setting. First, we need to estimate the value of a policy that is different than the one used to obtain the observations. Second, using a forward view such as (1) under off-policy sampling can cause it to be unstable, potentially resulting in divergence of the weights (Sutton et al., 2008). These issues can be avoided by constructing our off-policy algorithms to minimize a mean-squared projected Bellman error (MSPBE) with gradient descent (Sutton et al., 2009; Maei & Sutton, 2010; Maei, 2011).

The MSPBE was previously used to derive GTD(λ) (Maai, 2011), which is an online algorithm that can be used to learn off-policy predictions. GTD(λ) was not constructed to be exactly equivalent to any forward view and it is a natural question whether the algorithm can be improved from having such an equivalence, just as was the case with TD(λ) and true online TD(λ). In this section, we introduce an off-policy MSPBE and show how GTD(λ) can be derived. In the next section, we use the same MSPBE to construct a new off-policy forward view from which we will derive an exactly equivalent online backward view.

To obtain estimates for one distribution when the samples are generated under another distribution, we can weight the observations by the relative probabilities of these observations occurring under the target policy, as compared to the behavior distribution. This is called *importance sampling* (Rubinstein, 1981; Precup, Sutton & Singh, 2000). Recall that $b(a|s)$ and $\pi(a|s)$ denote the probabilities of selecting action a in state s according to the behavior policy and the target policy, respectively. After selecting an action A_t in a state S_t according to b , we observe a reward R_{t+1} . The expected value of this reward is $\mathbb{E}_b[R_{t+1}]$, but if we multiply the reward with the importance-sampling ratio $\rho_t \doteq \pi(A_t|S_t)/b(A_t|S_t)$ the expected value is

$$\begin{aligned}
\mathbb{E}_b[\rho_t R_{t+1} | S_t] &= \sum_a b(a|S_t) \frac{\pi(a|S_t)}{b(a|S_t)} \mathbb{E}[R_{t+1} | S_t, A_t = a] \\
&= \sum_a \pi(a|S_t) \mathbb{E}[R_{t+1} | S_t, A_t = a] \\
&= \mathbb{E}_\pi[R_{t+1} | S_t].
\end{aligned}$$

Therefore $\rho_t R_{t+1}$ is an unbiased sample for the reward under the target policy. This technique can be applied to all the rewards and value estimates in a given λ -return.

For instance, if we want to obtain an unbiased sample for the reward under the target policy n steps after the current state S_t , the total weight applied to this reward should be $\rho_t \rho_{t+1} \cdots \rho_{t+n-1}$. An off-policy λ -return starting from state S_t is given by

$$G_t^{\lambda\rho}(\boldsymbol{\theta}) = \rho_t \left(R_{t+1} + \gamma_{k+1}(1 - \lambda_{k+1})\boldsymbol{\phi}_{k+1}^\top \boldsymbol{\theta} + \gamma_{k+1}\lambda_{k+1}G_{k+1}^\lambda(\boldsymbol{\theta}) \right). \quad (6)$$

In contrast to G_t^λ , this return is defined as a function of a single weight vector $\boldsymbol{\theta}$. This is useful later, when we wish to determine the gradient of this return with respect to $\boldsymbol{\theta}$.

When using function approximation it is generally not possible to estimate the value of each state with full accuracy or, equivalently, to reduce the conditional expected TD error for each state to zero at the same time. More formally, let v_θ be a parameterized value function defined by $v_\theta(s) = \boldsymbol{\theta}^\top \boldsymbol{\phi}(s)$ and let T_π^λ be a parametrized Bellman operator defined, for any $v : \{s\} \rightarrow \mathbb{R}$, by

$$(T_\pi^\lambda v)(s) = \mathbb{E}_\pi [R_1 + \gamma_1(1 - \lambda_1)v(S_1) + \gamma_1\lambda_1(T_\pi^\lambda v)(S_1) \mid S_0 = s].$$

In general, we then cannot achieve $v_\theta = T_\pi^\lambda v_\theta$, because $T_\pi^\lambda v_\theta$ is not guaranteed to be a function that we can represent with our chosen function approximation. It is, however, possible to find the fixed point defined by

$$v_\theta = \Pi T_\pi^\lambda v_\theta. \quad (7)$$

where Πv is a projection of v into the space of representable functions $\{v_\theta \mid \boldsymbol{\theta} \in \mathbb{R}^n\}$. Let d be the steady-state distribution of states under the behavior policy. The projection of any v is then defined by

$$\Pi v = v_{\boldsymbol{\theta}_v}, \quad \text{where} \quad \boldsymbol{\theta}_v = \arg \min_{\boldsymbol{\theta}} \|v_\theta - v\|_d^2,$$

where $\|\cdot\|_d^2$ is a norm defined by $\|f\|_d^2 \doteq \sum_s d(s)f(s)^2$. Following Maei (2011), the projection is defined in terms of the steady-state distribution resulting from the behavior policy, which means that $d(s) = \lim_{t \rightarrow \infty} \mathbb{P}(S_t = s \mid A_j \sim b(\cdot|S_j), \forall j)$. This implies our objective weights the importance of the accuracy of the prediction in each state according to the relative frequency that this state occurs under the behavior policy, which is a natural choice for online learning.

The fixed point in (7) can be found by minimizing the MSPBE defined by (Maei, 2011)

$$J(\boldsymbol{\theta}) = \|v_\theta - \Pi T_\pi^\lambda v_\theta\|_d^2 = \mathbb{E}_b [\delta_k^\pi(\boldsymbol{\theta})\boldsymbol{\phi}_k]^\top \mathbb{E}_b [\boldsymbol{\phi}_k\boldsymbol{\phi}_k^\top]^{-1} \mathbb{E}_b [\delta_k^\pi(\boldsymbol{\theta})\boldsymbol{\phi}_k], \quad (8)$$

where $\delta_k^\pi(\boldsymbol{\theta}) \doteq (T_\pi^\lambda v_\theta)(S_k) - v_\theta(S_k)$ and where the expectancies are with respect to the steady-state distribution

d , as induced by the behavior policy b . The ideal gradient update for time step k is then

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \frac{1}{2} \alpha \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})|_{\boldsymbol{\theta}_k}, \quad (9)$$

where

$$\begin{aligned} & -\frac{1}{2} \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})|_{\boldsymbol{\theta}_k} \\ &= -\mathbb{E}_b [\nabla_{\boldsymbol{\theta}} \delta_k^\pi(\boldsymbol{\theta})\boldsymbol{\phi}_k^\top] \mathbb{E}_b [\boldsymbol{\phi}_k\boldsymbol{\phi}_k^\top]^{-1} \mathbb{E}_b [\delta_k^\pi(\boldsymbol{\theta}_k)\boldsymbol{\phi}_k] \\ &= \mathbb{E}_b [(\boldsymbol{\phi}_k - \nabla_{\boldsymbol{\theta}} G_k^{\lambda\rho}(\boldsymbol{\theta}))\boldsymbol{\phi}_k^\top] \mathbb{E}_b [\boldsymbol{\phi}_k\boldsymbol{\phi}_k^\top]^{-1} \mathbb{E}_b [\delta_k^\pi(\boldsymbol{\theta}_k)\boldsymbol{\phi}_k] \\ &= \mathbb{E}_b [\delta_k^\pi(\boldsymbol{\theta}_k)\boldsymbol{\phi}_k] \\ &\quad - \mathbb{E}_b [\nabla_{\boldsymbol{\theta}} G_k^{\lambda\rho}(\boldsymbol{\theta})\boldsymbol{\phi}_k^\top] \mathbb{E}_b [\boldsymbol{\phi}_k\boldsymbol{\phi}_k^\top]^{-1} \mathbb{E}_b [\delta_k^\pi(\boldsymbol{\theta}_k)\boldsymbol{\phi}_k] \\ &= \mathbb{E}_b [\delta_k^\pi(\boldsymbol{\theta}_k)\boldsymbol{\phi}_k] - \mathbb{E}_b [\nabla_{\boldsymbol{\theta}} G_k^{\lambda\rho}(\boldsymbol{\theta})\boldsymbol{\phi}_k]^\top \mathbf{w}_*, \end{aligned} \quad (10)$$

with $G_k^{\lambda\rho}$ as defined in (6), and where

$$\mathbf{w}_* \doteq \mathbb{E}_b [\boldsymbol{\phi}_k\boldsymbol{\phi}_k^\top]^{-1} \mathbb{E}_b [\delta_k^\pi(\boldsymbol{\theta}_k)\boldsymbol{\phi}_k].$$

Update (9) can be interpreted as an expected forward view.

The derivation of the GTD(λ) algorithm proceeds by exploiting the expected equivalences (Maei, 2011)

$$\begin{aligned} & \mathbb{E}_b [\nabla_{\boldsymbol{\theta}} G_k(\boldsymbol{\theta})\boldsymbol{\phi}_k^\top] \\ &= \mathbb{E}_b [\rho_k \gamma_{k+1}(1 - \lambda_{k+1})\boldsymbol{\phi}_{k+1}\boldsymbol{\phi}_k^\top] \\ &\quad + \mathbb{E}_b [\rho_k \gamma_{k+1}\lambda_{k+1} \nabla_{\boldsymbol{\theta}} G_{k+1}(\boldsymbol{\theta})\boldsymbol{\phi}_k^\top] \\ &= \mathbb{E}_b [\rho_k \gamma_{k+1}(1 - \lambda_{k+1})\boldsymbol{\phi}_{k+1}\boldsymbol{\phi}_k^\top] \\ &\quad + \mathbb{E}_b [\rho_{k-1}\gamma_k \lambda_k \nabla_{\boldsymbol{\theta}} G_k(\boldsymbol{\theta})\boldsymbol{\phi}_{k-1}^\top] \\ &= \mathbb{E}_b [\rho_k \gamma_{k+1}(1 - \lambda_{k+1})\boldsymbol{\phi}_{k+1}\boldsymbol{\phi}_k^\top] \\ &\quad + \mathbb{E}_b [\rho_{k-1}\gamma_k \lambda_k \rho_k \gamma_{k+1}(1 - \lambda_{k+1})\boldsymbol{\phi}_{k+1}\boldsymbol{\phi}_{k-1}^\top] \\ &\quad + \mathbb{E}_b [\rho_{k-1}\gamma_k \lambda_k \rho_k \gamma_{k+1}\lambda_{k+1} \nabla_{\boldsymbol{\theta}} G_{k+1}(\boldsymbol{\theta})\boldsymbol{\phi}_{k-1}^\top] \\ &= \mathbb{E}_b [\rho_k \gamma_{k+1}(1 - \lambda_{k+1})\boldsymbol{\phi}_{k+1}\boldsymbol{\phi}_k^\top] \\ &\quad + \mathbb{E}_b [\rho_{k-1}\gamma_k \lambda_k \rho_k \gamma_{k+1}(1 - \lambda_{k+1})\boldsymbol{\phi}_{k+1}\boldsymbol{\phi}_{k-1}^\top] \\ &\quad + \mathbb{E}_b [\rho_{k-2}\gamma_{k-1}\lambda_{k-1}\rho_{k-1}\gamma_k \lambda_k \nabla_{\boldsymbol{\theta}} G_k(\boldsymbol{\theta})\boldsymbol{\phi}_{k-2}^\top] \\ &= \dots \text{(Repeat until we reach } \boldsymbol{\phi}_0) \\ &= \mathbb{E}_b \left[\gamma_{k+1}(1 - \lambda_{k+1})\boldsymbol{\phi}_{k+1} \underbrace{\rho_k \sum_{j=0}^k \left(\prod_{i=j+1}^k \rho_{i-1}\gamma_i \lambda_i \right)}_{\doteq (\mathbf{e}_k^\nabla)^\top} \boldsymbol{\phi}_j^\top \right] \\ &= \mathbb{E}_b [\gamma_{k+1}(1 - \lambda_{k+1})\boldsymbol{\phi}_{k+1}(\mathbf{e}_k^\nabla)^\top], \end{aligned} \quad (11)$$

and, similarly, $\mathbb{E}_b [\delta_k^\pi(\boldsymbol{\theta}_k)\boldsymbol{\phi}_k] = \mathbb{E}_b [\delta_k(\boldsymbol{\theta}_k)\mathbf{e}_k^\nabla]$, where

$$\begin{aligned} \mathbf{e}_t^\nabla &= \rho_t(\gamma_t \lambda_t \mathbf{e}_{t-1}^\nabla + \boldsymbol{\phi}_t), \\ \delta_t(\boldsymbol{\theta}) &= R_{t+1} + \gamma_{t+1}\boldsymbol{\phi}_{t+1}^\top \boldsymbol{\theta} - \boldsymbol{\phi}_t^\top \boldsymbol{\theta}. \end{aligned} \quad (12)$$

The auxiliary vector $\mathbf{w}_t \approx \mathbf{w}_*$ can be updated with least mean squares (LMS) (Sutton et al., 2009; Maei, 2011), using the sample $\delta_t(\boldsymbol{\theta}_t)\mathbf{e}_t^\nabla \approx \mathbb{E}_b [\delta_t(\boldsymbol{\theta}_t)\mathbf{e}_t^\nabla] = \mathbb{E}_b [\delta_t^\pi(\boldsymbol{\theta}_t)\boldsymbol{\phi}_t]$

and the update

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \beta_t \delta_t (\boldsymbol{\theta}_t) \mathbf{e}_t^\nabla - \beta_t \phi_t^\top \mathbf{w}_t \phi_t.$$

The complete GTD(λ) algorithm is then defined by¹

$$\begin{aligned} \delta_t &= R_{t+1} + \gamma_{t+1} \phi_{t+1}^\top \boldsymbol{\theta}_t - \phi_t^\top \boldsymbol{\theta}_t, \\ \mathbf{e}_t^\nabla &= \rho_t (\gamma_t \lambda_t \mathbf{e}_{t-1}^\nabla + \phi_t), \\ \boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha_t \delta_t \mathbf{e}_t^\nabla - \alpha_t \gamma_{t+1} (1 - \lambda_{t+1}) \mathbf{w}_t^\top \mathbf{e}_t^\nabla \phi_{t+1}, \\ \mathbf{w}_{t+1} &= \mathbf{w}_t + \beta_t \delta_t \mathbf{e}_t^\nabla - \beta_t \phi_t^\top \mathbf{w}_t \phi_t. \end{aligned}$$

5 An off-policy forward view

In this section, we define an off-policy forward view which we turn into a fully equivalent backward view in the next section, using Theorem 1. GTD(λ) is derived by first turning an expected forward view into an expected backward view, and then sampling. We propose instead to sample the expected forward view directly and then invert the sampled forward view into an equivalent online backward view. This way we obtain an exact equivalence between forward and backward views instead of the expected equivalence of GTD(λ). This was previously not known to be possible, but it has the advantage that we can use the precise (potentially discounted and bootstrapped) sample returns consisting of all future rewards and state values in each update. This can result in more accurate predictions, as confirmed by our experiments in Section 7.

The new forward view derives from the MSPBE, as defined in (8), and more specifically from the gradient update defined by (9) and (10). To find an implementable interim forward view, we need sampled estimates of all three parts in (10). We discuss each of these parts separately.

Our interim forward view is defined in terms of a data horizon t , so the gradient of the MSPBE is taken to $\boldsymbol{\theta}_k^t$ rather than $\boldsymbol{\theta}_k$. Furthermore, δ_k^π is defined as the error between a λ -return and a current estimate, and therefore we need to construct an interim λ -return. To estimate the first term of (10) we therefore need an estimate for $\mathbb{E}_b[\delta_k^\pi(\boldsymbol{\theta}_k^t) \phi_k] = \mathbb{E}_b[G_{k,t}^{\lambda\rho} - \phi_k^\top \boldsymbol{\theta}_k^t]$, for some suitably defined $G_{k,t}^{\lambda\rho}$.

The variance of off-policy updates is often lower when we weight the errors (that is, the difference between the return and the current estimate) with the importance-sampling ratios, rather than weighting the returns (Sutton et al., 2014). Let $\delta_k = R_{k+1} + \gamma_{k+1} \phi_{k+1}^\top \boldsymbol{\theta}_k - \phi_k^\top \boldsymbol{\theta}_{k-1}$ denote a one-step TD error. The on-policy return used in the forward view (1) can then be written as a sum of such errors:

$$G_{k,t}^\lambda = \phi_k^\top \boldsymbol{\theta}_{k-1} + \sum_{j=k}^{t-1} \left(\prod_{i=k+1}^j \gamma_i \lambda_i \right) \delta_j.$$

¹Dann, Neumann and Peters (2014) call this algorithm TDC(λ), but we use the original name by Maei (2011).

We apply the importance-sampling weights to the one-step TD errors, rather than just to the reward and bootstrapped value estimate.² This does not affect the expected value, because $\mathbb{E}_b[\rho_k \phi_k^\top \boldsymbol{\theta}_{k-1} | S_k] = \mathbb{E}_b[\phi_k^\top \boldsymbol{\theta}_{k-1} | S_k]$, but it can have a beneficial effect on the variance of the resulting updates. A sampled off-policy error is then

$$G_{k,t}^{\lambda\rho} - \rho_k \phi_k^\top \boldsymbol{\theta}_k^t \approx \mathbb{E}_b[\delta_k^\pi(\boldsymbol{\theta}_k^t) \phi_k], \quad (13)$$

where

$$G_{k,t}^{\lambda\rho} \doteq \rho_k \phi_k^\top \boldsymbol{\theta}_{k-1} + \rho_k \sum_{j=k}^{t-1} \left(\prod_{i=k+1}^j \gamma_i \lambda_i \rho_i \right) \delta_j.$$

An equivalent recursive definition for $G_{k,t}^{\lambda\rho}$ is

$$\begin{aligned} G_{k,t}^{\lambda\rho} &= \rho_k \left(R_{k+1} + \gamma_{k+1} (1 - \lambda_{k+1} \rho_{k+1}) \phi_{k+1}^\top \boldsymbol{\theta}_k \right. \\ &\quad \left. + \gamma_{k+1} \lambda_{k+1} G_{k+1,t}^{\lambda\rho} \right), \end{aligned} \quad (14)$$

for $k < t$, and $G_{t,t}^{\lambda\rho} \doteq \rho_t \phi_t^\top \boldsymbol{\theta}_{t-1}$. In the on-policy case, when $\rho_k = 1$ for all k , $G_{k,t}^{\lambda\rho}$ reduces exactly to $G_{k,t}^\lambda$, as used in the forward view (1) for true online TD(λ). Furthermore, $\mathbb{E}_b[G_{k,t}^{\lambda\rho} | S_k = s] = \mathbb{E}_\pi[G_{k,t}^\lambda | S_k = s]$ for any s .

For the second term in (10), which can be thought of as the gradient correction term, we need an estimate $\mathbf{w}_k \approx \mathbf{w}_*$. As in the derivation of GTD(λ), we use a LMS update. Assuming we have data up to t , the ideal forward-view update for \mathbf{w}_k^t is then

$$\mathbf{w}_{k+1}^t = \mathbf{w}_k^t + \beta_k (\delta_{k,t}^{\lambda\rho} - \phi_k^\top \mathbf{w}_k^t) \phi_k, \quad (15)$$

for some appropriate sample $\delta_{k,t}^{\lambda\rho} \approx \mathbb{E}_b[\delta_k^\pi(\boldsymbol{\theta}_k)]$. A natural interim estimate is defined by

$$\delta_{k,t}^{\lambda\rho} = \rho_k (\delta_k + \gamma \lambda \delta_{k+1,t}^{\lambda\rho}), \quad (16)$$

where $\delta_{t,t}^{\lambda\rho} = 0$ and

$$\delta_k = R_{k+1} + \gamma \boldsymbol{\theta}_k^\top \phi_{k+1} - \boldsymbol{\theta}_k^\top \phi_k.$$

This is not the only possible way to estimate \mathbf{w}_* , but this choice ensures the resulting algorithm is equivalent to GTD(0) when $\lambda = 0$, allowing us to investigate the effects of the true online equivalence and the resulting new trace updates in some isolation without having to worry about other potential differences between the algorithms. In the next section we construct an equivalent backward view for (15) to compute the sequence $\{\mathbf{w}_t\}$, where $\mathbf{w}_t = \mathbf{w}_t^t, \forall t$.

²For the PTD(λ) and PQ(λ) algorithms, Sutton et al. (2014) propose another weighting based on weighting flat return errors containing multiple rewards. In contrast, our weighting is chosen to be consistent with GTD(λ). True online versions of PTD(λ) and PQ(λ) exist, but we do not consider them further in this paper.

Finally, we use the expected equivalence proved in (11), and then sample to obtain

$$\gamma_{k+1}(1 - \lambda_{k+1})\phi_{k+1}(\mathbf{e}_k^\nabla)^\top \approx \mathbb{E}_b[\nabla_\theta G_k(\theta)\phi_k^\top], \quad (17)$$

with \mathbf{e}_k^∇ as defined in (12).

We now have all the pieces to state the off-policy forward view for θ . We approximate the expected forward view as defined by (9) and (10) by using the sampled estimates (13), (17) and $\mathbf{w}_k = \mathbf{w}_k^k \approx \mathbf{w}_*$, with \mathbf{w}_k^k as defined by (15). This gives us the interim forward view

$$\begin{aligned} \theta_{k+1}^t &= \theta_k^t + \alpha_k(G_{k,t}^{\lambda\rho} - \rho_k\phi_k^\top\theta_k^t)\phi_k \\ &\quad - \alpha_k\gamma_{k+1}(1 - \lambda_{k+1})\phi_{k+1}\mathbf{w}_k^\top\mathbf{e}_k^\nabla, \end{aligned} \quad (18)$$

with $G_{k,t}^{\lambda\rho}$ as defined in (14).

6 Backward view: true online GTD(λ)

In this section, we apply Theorem 1 to convert the off-policy forward view as given by (18) into an efficient online backward view. First, we consider \mathbf{w} .

Theorem 3 (Auxiliary vectors). *The vector \mathbf{w}_t^t , as defined by the forward view in (15), is equal to \mathbf{w}_t as defined by the backward view*

$$\begin{aligned} \mathbf{e}_t^w &= \rho_{t-1}\gamma_t\lambda_t\mathbf{e}_{t-1}^w + \beta_t(1 - \rho_{t-1}\gamma_t\lambda_t\phi_t^\top\mathbf{e}_{t-1}^w)\phi_t, \\ \mathbf{w}_{t+1} &= \mathbf{w}_t + \rho_t\delta_t\mathbf{e}_t^w - \beta_t\phi_t^\top\mathbf{w}_t\phi_t, \end{aligned}$$

where $\mathbf{e}_0^w = \beta_0\phi_0$, $\mathbf{w}_0 = \mathbf{w}_0^t, \forall t$, and

$$\delta_t \doteq R_{t+1} + \gamma\phi_{t+1}^\top\theta_t - \phi_t^\top\theta_t.$$

Proof. We apply Theorem 1 by substituting $\theta_t = \mathbf{w}_t$, $\eta_t = \beta_t$, $\mathbf{x}_t = \mathbf{0}$ and $Y_k^t = \delta_{k,t}^{\lambda\rho}$, as defined in (16). Then

$$\delta_{k,t+1}^{\lambda\rho} - \delta_{k,t}^{\lambda\rho} = \rho_k\gamma_{k+1}\lambda_{k+1}(\delta_{k+1,t+1}^{\lambda\rho} - \delta_{k+1,t}^{\lambda\rho}),$$

which implies $c_k = \rho_k\gamma_{k+1}\lambda_{k+1}$. Finally, $Y_t^t = \delta_{t,t}^{\lambda\rho} = 0$ and $Y_t^{t+1} - Y_t^t = \delta_{t,t+1}^{\lambda\rho} = \rho_t\delta_t$. Inserting these substitutions into the backward view in Theorem 1 immediately yields the backward view in the current theorem. \square

Theorem 4 (True online GTD(λ)). *For any t , the weight vector θ_t^t as defined by forward view in (18) is equal to θ_t , as defined by the backward view*

$$\begin{aligned} \mathbf{e}_t &= \rho_t(\gamma_t\lambda_t\mathbf{e}_{t-1} + \alpha_t(1 - \rho_t\gamma_t\lambda_t\phi_t^\top\mathbf{e}_{t-1})\phi_t), \\ \mathbf{e}_t^\nabla &= \rho_t(\gamma_t\lambda_t\mathbf{e}_{t-1}^\nabla + \phi_t), \\ \theta_{t+1} &= \theta_t + \delta_t\mathbf{e}_t + (\mathbf{e}_t - \alpha_t\rho_t\phi_t)(\theta_t - \theta_{t-1})^\top\phi_t \\ &\quad - \alpha_t\gamma_{t+1}(1 - \lambda_{t+1})\mathbf{w}_t^\top\mathbf{e}_t^\nabla\phi_{t+1}, \end{aligned}$$

with \mathbf{w}_t and δ_t as defined in Theorem 3.

Proof. Again, we apply Theorem 1. Substitute $\eta_t = \rho_t\alpha_t$, $\mathbf{x}_k = -\alpha_k\gamma_{k+1}(1 - \lambda_{k+1})\phi_{k+1}\mathbf{w}_k^\top\mathbf{e}_k^\nabla$, $Y_t^t = \theta_{t-1}^\top\phi_t$ and

$$Y_k^t = R_{k+1} + \gamma_{k+1}(1 - \lambda_{k+1}\rho_{k+1})\theta_k^\top\phi_{k+1} + \gamma\lambda G_{k+1,t}^{\lambda\rho}.$$

This last substitution implies

$$Y_k^{t+1} - Y_k^t = \gamma_{k+1}\lambda_{k+1}\rho_{k+1}(Y_{k+1}^{t+1} - Y_{k+1}^t),$$

so that $c_k = \gamma_{k+1}\lambda_{k+1}\rho_{k+1}$. Furthermore,

$$\begin{aligned} Y_t^{t+1} - Y_t^t &= R_{t+1} + \gamma\theta_t^\top\phi_{t+1} - \theta_{t-1}^\top\phi_t \\ &= \delta_t + (\theta_t - \theta_{t-1})^\top\phi_t. \end{aligned}$$

Applying Theorem 1 with these substitutions, and replacing \mathbf{w}_t^t with the equivalent \mathbf{w}_t , yields the backward view

$$\begin{aligned} \theta_{t+1} &= \theta_t + (\delta_t + (\theta_t - \theta_{t-1})^\top\phi_t)\mathbf{e}_t \\ &\quad + \alpha_t\rho_t(\theta_{t-1}^\top\phi_t - \theta_t^\top\phi_t)\phi_t \\ &\quad - \alpha_t\gamma_{t+1}(1 - \lambda_{t+1})\mathbf{w}_t^\top\mathbf{e}_t^\nabla\phi_{t+1}, \\ &= \theta_t + \delta_t\mathbf{e}_t + (\mathbf{e}_t - \alpha_t\rho_t\phi_t)\phi_t^\top(\theta_t - \theta_{t-1}) \\ &\quad - \alpha_t\gamma_{t+1}(1 - \lambda_{t+1})\mathbf{w}_t^\top\mathbf{e}_t^\nabla\phi_{t+1}, \end{aligned}$$

where $\mathbf{e}_0 = \alpha_0\rho_0\phi_0$ and

$$\mathbf{e}_t = \rho_t\gamma_t\lambda_t\mathbf{e}_{t-1} + \alpha_t\rho_t(1 - \rho_t\gamma_t\lambda_t\mathbf{e}_{t-1}^\top\phi_t)\phi_t. \quad \square$$

True online GTD(λ) algorithm is then defined by

$$\begin{aligned} \delta_t &= R_{t+1} + \gamma\phi_{t+1}^\top\theta_t - \phi_t^\top\theta_t, \\ \mathbf{e}_t &= \rho_t(\gamma_t\lambda_t\mathbf{e}_{t-1} + \alpha_t(1 - \rho_t\gamma_t\lambda_t\phi_t^\top\mathbf{e}_{t-1})\phi_t), \\ \mathbf{e}_t^\nabla &= \rho_t(\gamma_t\lambda_t\mathbf{e}_{t-1}^\nabla + \phi_t), \\ \mathbf{e}_t^w &= \rho_{t-1}\gamma_t\lambda_t\mathbf{e}_{t-1}^w + \beta_t(1 - \rho_{t-1}\gamma_t\lambda_t\phi_t^\top\mathbf{e}_{t-1}^w)\phi_t, \\ \theta_{t+1} &= \theta_t + \delta_t\mathbf{e}_t + (\mathbf{e}_t - \alpha_t\rho_t\phi_t)(\theta_t - \theta_{t-1})^\top\phi_t \\ &\quad - \alpha_t\gamma_{t+1}(1 - \lambda_{t+1})\mathbf{w}_t^\top\mathbf{e}_t^\nabla\phi_{t+1}, \\ \mathbf{w}_{t+1} &= \mathbf{w}_t + \rho_t\delta_t\mathbf{e}_t^w - \beta_t\phi_t^\top\mathbf{w}_t\phi_t. \end{aligned}$$

The traces \mathbf{e}_t and \mathbf{e}_t^w are dutch traces. The trace \mathbf{e}_t^∇ is an accumulating trace that follows from the gradient correction, as discussed in Section 4. It might be possible to adapt the forward view to replace \mathbf{e}_t^∇ with \mathbf{e}_t . This is already possible in practice and in preliminary experiments the resulting algorithm performed similar to true online GTD(λ). A more detailed investigation of this possibility is left for future work.

For $\lambda = 0$ the algorithm reduces to

$$\begin{aligned} \theta_{t+1} &= \theta_t + \alpha_t\rho_t\delta_t\phi_t - \alpha_t\rho_t\gamma_{t+1}\mathbf{w}_t^\top\phi_t\phi_{t+1}, \\ \mathbf{w}_{t+1} &= \mathbf{w}_t + \beta_t\rho_t\delta_t\phi_t - \beta_t\phi_t^\top\mathbf{w}_t\phi_t, \end{aligned}$$

which is precisely GTD(0).³

³The on-policy variant of this algorithm, with $\rho_t = 1$ for all t , is known as TDC (Sutton et al., 2009; Maei, 2011).

7 Experiments

We compare true online GTD(λ) to GTD(λ) empirically in various settings. The main goal of the experiments is to test the intuition that true online GTD(λ) should be more robust to high step sizes and high λ , due to its true online equivalence and better behaved traces. This was shown to be the case for true online TD(λ) (van Seijen & Sutton, 2014), and the experiments serve to verify that this extends to the off-policy setting with true online GTD(λ). This is relevant because it implies true online GTD(λ) should then be easier to tune in practice, and because these parameters can effect the limiting performance of the algorithms as well.

Both algorithms optimize the MSPBE, as given in (8), which is a function of λ . When the state representation is of poor quality, the solution that minimizes the MSPBE can still have a high mean-squared error (MSE): $\|v_\theta - v^\pi\|_d^2$. This means that with a low λ we are not always guaranteed to reach a low MSE, even asymptotically. The closer λ is to one, the closer the MSPBE becomes to the MSE, with equality for $\lambda = 1$. In practice this implies that sometimes we need a high λ to be able to obtain an sufficiently accurate predictions, even if we run the algorithms a long time.

To illustrate these points, we investigate a fairly simple problem. The problem setting is a random walk consisting of 15 states that can be thought to lie on a horizontal line. In each state we have two actions: move one state to the left, or one state to the right. If we move left in the left-most state, s_1 , we bounce back into that state. If we move right in the right-most state, s_{15} , the episode ends and we get a reward of +1. On all other time steps, the reward is zero. Each episode starts in s_1 , which is the left-most state.

This problem setting is similar to the one used by van Seijen and Sutton (2014), with three differences. First, we use 15 rather than 11 states, but this makes little difference to the conclusions. Second, we turn it into an off-policy learning problem, as we describe in a moment. Third, we use different state representations. This last point is because we want to test the performance of the algorithm not just with features that can accurately represent the value function, as used by van Seijen and Sutton, but also with features that cannot reduce the MSE all the way to zero.

In the original problem, there was a 0.9 probability of moving right in each state (van Seijen & Sutton, 2014). Here, we interpret these probabilities as begin due to a behavior policy that selects the ‘right’ action with probability 0.9. Then, we formulate a target policy that want to move right more often, with probability 0.95. The stochastic target policy demonstrates that our algorithm is applicable to arbitrary off-policy learning tasks, and that the results do not depend on the target policy being deterministic. We did also test the performance for a deterministic policy that moves right always and the results are similar to those given

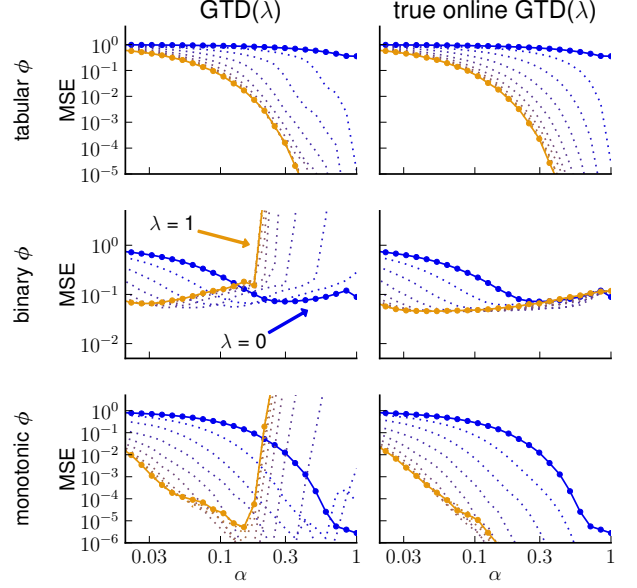


Figure 1: The MSE on the random walk of GTD(λ) (left column) and true online GTD(λ) (right column). The x -axis shows α , and the different lines are for different λ , with $\lambda = 0$ in blue and $\lambda = 1$ in orange. The top row is for 15 tabular features, the middle row for 4 binary features, and the bottom row for 2 monotonic features. The MSE is minimized over β .

below. Because this is an episodic task, $\gamma = 1$.

As stated above, we define three different state representations. In the first task, we use tabular features, such that $\phi(s_i)$ is a vector of 15 elements, with the i th element equal to one and all other elements equal to zero. In the second task the state number is turned into a binary representation, such that $\phi(s_1) = (0, 0, 0, 1)^\top$, $\phi(s_2) = (0, 0, 1, 0)^\top$, $\phi(s_3) = (0, 0, 1, 1)^\top$, and so on up to $\phi(s_{15}) = (1, 1, 1, 1)^\top$. The features are then normalized to be unit vectors, such that for instance $\phi(s_3) = (0, 0, \frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})^\top$ and $\phi(s_{15}) = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2})^\top$. In our final representation, we use one monotonically increasing feature and one monotonically decreasing feature, such that $\phi(s_i) = (\frac{14-i+1}{14}, \frac{i-1}{14})^\top$ for all i . These features were not normalized.

For α the range of parameters was from 2^{-8} to 1 with steps in the exponent of 0.25 so that $\alpha \in \{2^{-8}, 2^{-7.75}, \dots, 1\}$. The secondary step size β was varied over the same range, with the addition of $\beta = 0$. The trace parameter λ was varied from 0 to $1 - 2^{-10} \approx 0.999$ with steps of -1 in the exponent and with the addition of $\lambda = 1$, such that $\lambda \in \{0, 1 - 2^{-1}, \dots, 1 - 2^{-9}, 1 - 2^{-10}, 1\}$.

The MSE (averaged over 20 repetitions) after 10 episodes for all three representations are shown in Figure 1. The left graphs all correspond to GTD(λ) and the plots on the right

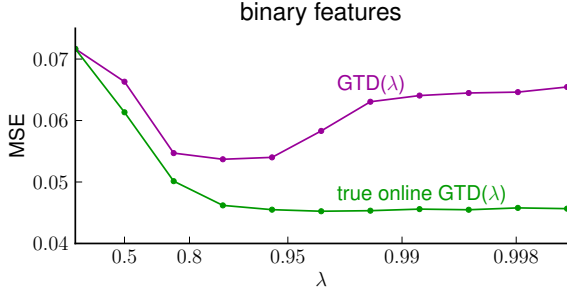


Figure 2: The MSE on the random walk for different λ of GTD(λ) and true online GTD(λ) for optimized α and β and binary features.

are for true online GTD(λ). Each graph show the MSE as a function of α , with different lines for different values of λ of which the extremes are highlighted ($\lambda = 0$ is blue; $\lambda = 1$ is orange). In all cases, the MSE was minimized for β , but this secondary step size had little impact on the performance at all in these problems. Note that the blue lines in the pair of graphs in each row are exactly equal, because by design the algorithms are equivalent for $\lambda = 0$.

In the top plots, the tabular representation was used and we see that especially with high λ both algorithms reach low prediction errors. This demonstrates that indeed learning can be faster with higher λ . When using function approximation, in the middle and bottom graphs, the benefit of having an online equivalence to a well-defined forward view becomes apparent. For both representations, the performance of GTD(λ) with higher λ begins to deteriorate around $\alpha = 0.2$. In contrast, true online GTD(λ) performs well even for $\alpha = \lambda = 1$. Note the log scale of the y -axis; the difference in MSE is many orders of magnitude.

In practice it is not always possible to fully tune the algorithmic parameters and therefore the robustness of true online GTD(λ) to different settings is important. However, it is still interesting to see what the best performance could be for a fully tuned algorithm. Therefore, in Figure 2 we show the MSE as a function of λ when minimized over both α and β . For all λ , true online GTD(λ) outperforms GTD(λ).

8 Discussion

The main theoretical contribution of this paper is a general theorem for equivalences between forward and backward views. The theorem allows us to find an efficient fully equivalent online algorithm for a desired forward view. The theorem is as general as required and as specific as possible for all applications of it in this paper, and in its current form it is limited to forward views for which an $O(n)$ backward view exists. The theorem can be generalized further, to include recursive (off-policy) LSTD(λ) (Boyan, 1999) and other algorithms that can be formulated in terms of forward

views (cf. Geist & Scherrer, 2014; Dann, Neumann & Peters, 2014), but we did not investigate these extensions.

We used Theorem 1 to construct a new off-policy algorithm named true online GTD(λ), which is the first TD algorithm to have an exact online equivalence to a off-policy forward view. We constructed this forward view to maintain equivalence to the existing GTD(λ) algorithm for $\lambda = 0$. The forward view we proposed is not the only possible, and in particular it will be interesting to investigate different methods of importance sampling. We could for instance use the importance sampling as proposed by Sutton et al. (2014). We did construct the resulting online algorithm, and in preliminary tests its performance was similar to true online GTD(λ). Likewise, if desired, it is possible to obtain a full online equivalence to off-policy Monte Carlo for $\lambda = 1$ by constructing a forward view that achieves this. For instance we could use a similar forward view as used in the paper, but then apply the importance-sampling ratios only to the returns rather than to the errors. For now, it remains an open question what the best off-policy forward view is.

True online GTD(λ) is limited to state-value estimates. It is straightforward to construct a corresponding algorithm for action values, similar to the correspondence between GTD(λ) and GQ(λ) (Maei & Sutton, 2010; Maei, 2011) and between PTD(λ) and PQ(λ) (Sutton et al., 2014). We leave such an extension for future work.

Acknowledgments

The authors thank Joseph Modayil, Harm van Seijen and Adam White for fruitful discussions that helped improve the quality of this work. This work was supported by grants from Alberta Innovates – Technology Futures, the National Science and Engineering Research Council of Canada, and the Alberta Innovates Centre for Machine Learning.

References

- Boyan, J. A., (1999). Least-squares temporal difference learning. In *Proceedings of the 16th International Conference on Machine Learning*, pp. 49–56.
- Dann, C., Neumann, G., & Peters, J. (2014). Policy evaluation with temporal differences: A survey and comparison. In *Journal of Machine Learning Research* 15:809–883.
- Geist, M., & Scherrer, B. (2014). Off-policy learning with eligibility traces: A survey. In *Journal of Machine Learning Research* 15:289–333.
- Maei, H. R., & Sutton, R. S. (2010). GQ(λ): A general gradient algorithm for temporal-difference prediction learning with eligibility traces. In *Proceedings of the Third Conference on Artificial General Intelligence*, pp. 91–96. Atlantis Press.

- Maei, H. R. (2011). *Gradient Temporal-Difference Learning Algorithms*. PhD thesis, University of Alberta.
- Precup, D., Sutton, R. S., & Singh, S. (2000). Eligibility traces for off-policy policy evaluation. In *Proceedings of the 17th International Conference on Machine Learning*, pp. 759–766. Morgan Kaufmann.
- Rubinstein, R. Y. (1981). *Simulation and the Monte Carlo method*. New York, Wiley.
- Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine Learning* 3:9–44.
- Sutton, R. S., Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. MIT Press.
- Sutton, R. S., Mahmood, A. R. , Precup, D., & van Hasselt, H. (2014). A new $Q(\lambda)$ with interim forward view and Monte Carlo equivalence. In *Proceedings of the 31st International Conference on Machine Learning*. JMLR W&CP 32(2).
- Sutton, R. S., Maei, H. R., Precup, D., Bhatnagar, S., Silver, D., Szepesvári, Cs., & Wiewiora, E. (2009). Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 993–1000, ACM.
- Sutton, R. S., Modayil, J., Delp, M., Degris, T., Pilarski, P. M., White, A., & Precup, D. (2011). Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems*, pp. 761–768.
- Sutton, R. S., Szepesvári, Cs., & Maei, H. R. (2008). A convergent $O(n)$ algorithm for off-policy temporal-difference learning with linear function approximation. In *Advances in Neural Information Processing Systems 21*, pp. 1609–1616. MIT Press.
- van Seijen, H., & Sutton, R. S. (2014). True online TD(λ). In *Proceedings of the 31st International Conference on Machine Learning*. JMLR W&CP 32(1):692–700.

Constraint-based Causal Discovery: Conflict Resolution with Answer Set Programming

Antti Hyttinen and **Frederick Eberhardt**
California Institute of Technology
Pasadena, CA, USA

Matti Järvisalo
HIIT & Department of Computer Science
University of Helsinki, Finland

Abstract

Recent approaches to causal discovery based on Boolean satisfiability solvers have opened new opportunities to consider search spaces for causal models with both feedback cycles and unmeasured confounders. However, the available methods have so far not been able to provide a principled account of how to handle conflicting constraints that arise from statistical variability. Here we present a new approach that preserves the versatility of Boolean constraint solving *and* attains a high accuracy despite the presence of statistical errors. We develop a new logical encoding of (in)dependence constraints that is both well suited for the domain and allows for faster solving. We represent this encoding in Answer Set Programming (ASP), and apply a state-of-the-art ASP solver for the optimization task. Based on different theoretical motivations, we explore a variety of methods to handle statistical errors. Our approach currently scales to cyclic latent variable models with up to seven observed variables and outperforms the available constraint-based methods in accuracy.

1 INTRODUCTION

The search for causal relations underlies many scientific fields. Unlike mere correlational information, causal relations support predictions of how a system will behave when it is subject to an intervention. In the causal Bayes net framework (Spirtes et al., 1993; Pearl, 2000) the causal structure is represented in terms of a directed graph (see Figure 1). One of the most widely applicable approaches to discovering the causal structure uses independence and dependence constraints obtained from statistical tests to narrow down the candidate graphs that may have produced the data. Such an inference relies on the now well-understood assumptions of *causal Markov* and *causal*

faithfulness (Spirtes et al., 1993). Unlike many other approaches, these constraint-based causal discovery methods can allow for the presence of latent confounders, feedback cycles and the utilisation of several (partially overlapping) observational or experimental data sets.

Even without experimentation (or additional assumptions, such as time order), and despite the generality of the model space, constraint-based methods can infer some causal orientations on the basis of *v-structures* (unshielded colliders). A *v-structure* in a graph is a triple of variables, such as $\langle x, z, y \rangle$ in Figure 1, where z is a common child of x and y , but x and y are non-adjacent in the graph. *V-structures* can be identified because of the specific (in)dependence relations they imply (here, $x \not\perp\!\!\!\perp z$, $z \not\perp\!\!\!\perp y$ and $x \perp\!\!\!\perp y$ are jointly sufficient to identify the *v-structure*). The edges that are thus oriented provide the basis for all further orientation inferences in constraint-based algorithms such as PC and FCI (Spirtes et al., 1993); e.g. identifying the *v-structure* in Figure 1 enables the additional orientation of the zw -edge. However, when processing sample data, the above inference is often prone to error. Establishing the further dependence $x \not\perp\!\!\!\perp y \mid z$ would confirm the inference. If this dependence does not hold, we have a case of a *conflict*: There is no causal graph that satisfies all available (in)dependence constraints (while respecting Markov and faithfulness).

The problem of conflicting constraints is exacerbated when trying to integrate multiple observational and experimental data sets in which the sets of measured variables *overlap* only partially. Unlike the case for one passively observed data set, the characterization of the class of graphs consistent with the (in)dependence constraints is more difficult in this setting: the graphs may disagree on orientations, adjacencies, and ancestral relations (Tsamardinos et al., 2012). Triantafillou et al. (2010) and Hyttinen et al. (2013) have started using general Boolean satisfiability (SAT) solvers (Biere et al., 2009) to integrate the various constraints. The basic idea of these methods is to convert the (in)dependence constraints found in the data into logical constraints on the presence and absence of certain pathways in the underlying causal structure, and to use a

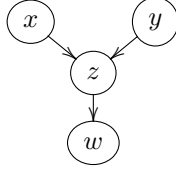


Figure 1: Example causal graph (see text for details).

SAT-solver to find the causal structures consistent with all constraints.

For pure SAT methods, conflicted constraints imply unsatisfiability. For these methods to work at all in practice, conflict handling is paramount. We address this problem at a variety of levels. We represent the task as a constraint optimization problem, and discuss in Section 3 different weighting schemes of the constraints and the theoretical motivations that support them. Then, we present in Section 5 a new encoding that relates (in)dependence constraints directly to one another via the operations of *intervening*, *conditioning* and *marginalization*. The encoding naturally captures the central ideas of constraint integration as it more directly connects constraints that refer to the same graphical neighborhood. We represent this encoding in the constraint optimization paradigm of Answer Set Programming (ASP) (Gelfond and Lifschitz, 1988; Niemelä, 1999; Simons et al., 2002). Finally, we compare the reliability of our proposed methods with available existing algorithms in Section 7.

2 PRELIMINARIES

For notational simplicity, we restrict the presentation to the setting with a single passive observational data set. However, the approach extends naturally to the general case of multiple overlapping experimental data sets; details are provided in the supplement.

We consider the class \mathcal{G} of *causal graphs* of the form $G = (\mathbf{V}, \mathbf{E})$, where \mathbf{V} is the set of nodes (associated with random variables) of G , and the edge relation $\mathbf{E} = \mathbf{E}_{\rightarrow} \cup \mathbf{E}_{\leftrightarrow}$ is composed of a set \mathbf{E}_{\rightarrow} of directed edges and a set $\mathbf{E}_{\leftrightarrow}$ of bi-directed edges. A bi-directed edge \leftrightarrow represents a *latent confounder*, i.e. an unmeasured common cause of two or more of the observed variables. In other words, we allow for the presence of feedback cycles and do not assume causal sufficiency. We define a *path* as a sequence of consecutive edges in the graph, without any restrictions on the types or orientations of the edges involved. A vertex appears as a *collider* on a path if both its adjacent edges on the path point into the vertex.

(In)dependence constraints can be brought into correspondence with structural properties of the graph using the d-separation criterion (Pearl, 2000). A path in graph G is *d-connecting* with respect to a conditioning set \mathbf{C} if every

collider c on the path is in \mathbf{C} and no other nodes on the path are in \mathbf{C} ; otherwise the path is *d-separated* (or “blocked”). A pair of nodes are d-connected given a conditioning set \mathbf{C} if there is at least one d-connecting path between them; otherwise they are d-separated.¹ Under the causal Markov and faithfulness assumptions, two variables x and y are independent conditional on a set of variables \mathbf{C} iff x and y are d-separated given \mathbf{C} in the graph G .²

Let \mathbf{D} be an i.i.d. data set sampled from a distribution that is Markov and faithful to an underlying “true” causal graph $G_t = (\mathbf{V}, \mathbf{E})$. Overall, the aim of causal discovery is to recover as many properties of G_t as possible from the data \mathbf{D} . There are a variety of ways to proceed. In light of the generality of the search space we consider, we focus on the graphical constraints implied by the conditional (in)dependencies found in the data. Our proposal is that the causal discovery problem is addressed well by solutions to the following abstract constrained optimization problem.

Problem Statement

INPUT: A set \mathbf{K} of conditional independence and dependence constraints over a set \mathbf{V} of variables, and a non-negative weight (cost) $w(k)$ for each $k \in \mathbf{K}$.

TASK: Find a causal graph G^* over the vertex set \mathbf{V} such that

$$G^* \in \arg \min_{G \in \mathcal{G}} \sum_{k \in \mathbf{K} : G \not\models k} w(k). \quad (1)$$

In words, our goal is to find a single representative graph G^* that minimizes the sum of the weights of the given conditional independence and dependence constraints *not* implied by G^* . The idea is that the constraints can be weighted according to their reliability, and that conflicts among the constraints are well-resolved when the sum of the weights of the constraints not satisfied by the output graph are minimized. We take the set \mathbf{K} to be the set of all (in)dependence constraints testable in the data \mathbf{D} .³

This formalization poses two key challenges: 1) How to define the weighting scheme $w(k)$ such that the solutions (causal graphs) are “as similar as possible” to the true graph? 2) How to actually find a representative graph G^* , i.e., how to represent the constraints efficiently such that one can minimize the objective function (an NP-hard optimization problem) defined above? In the following, we will

¹This definition is equivalent to Pearl’s standard definition (Studený, 1998; Koster, 2002).

²See Spirtes (1995), Pearl and Dechter (1996) and Neal (2000) for discussions of d-separation in cyclic models.

³When considering experimental data sets, the weights of (in)dependence constraints of the *manipulated* distributions enter into the minimization as well; see Appendix A.

propose solutions to both of these challenges. We note that once a representative graph G^* has been found, the independence oracle methods of Hyttinen et al. (2013) can be used to understand the common properties, such as adjacencies and ancestral relations, of all solutions that satisfy the same set of constraints.

3 PREVIOUS WORK

Statistical variability results in incorrect test results, but such errors are only detectable when the test results have conflicting implications. Conflicts can arise directly between test results or, more commonly, in combination with search space assumptions. Whether or not conflicts are detected depends on the tests that have been performed. Thus, one way of “handling” conflicts is to avoid performing tests whose results can conflict, or not to draw the full set of possible inferences from a set of test results. This is what the standard PC-algorithm does; it basically avoids conflicts.⁴ This approach proved to be unreliable for the detection of v-structures, so a *conservative* extension was added that checks additional tests to verify the inference to the v-structures (see example in Section 1). The resulting cPC-algorithm (Ramsey et al., 2006) marks any conflicted v-structures and abstains from the orientation inference, i.e. it returns a “don’t know” for this part of the graph. The FCI and cFCI algorithms are analogous in this regard to PC and cPC, respectively, only that the search space is enlarged to allow for latent confounders. A more generic approach, not focused on v-structures, reduces the occurrence of conflicts by controlling which test results are accepted. Li and Wang (2009) control the false discovery rate of the tests that PC performs, while Tsamardinos et al. (2012) use different p-value thresholds to infer independence vs. dependence constraints. Very recently, Triantafillou and Tsamardinos (2014) developed a scheme that uses the p-values from tests performed by FCI to rank (a specific class of) structural constraints on the underlying graph, and then use a SAT-based procedure to satisfy as many constraints as possible. The selection of tests by FCI depends on the results of earlier tests. Thus the conflict resolution (in terms of the ranked constraints) only handles conflicts between tests that were selected. This scales very well, but the theoretical account of the accuracy of the output model is unclear, as the selection of tests interacts with the conflict resolution.

Score-based algorithms take a very different approach to inconsistencies in data (Cooper and Herskovits, 1992; Chickering, 2002). Instead of explicitly determining the (in)dependence constraints (and finding them to be inconsistent) the conflicts are implicitly resolved by a direct integration of the data points into a score that identifies the

graph (or equivalence class) that maximizes the Bayesian posterior. Such an approach provides a clear theoretical account in which sense the output is “closest” to the true graph (equivalence class). Although mostly restricted to DAGs, Claassen and Heskes (2012) have transferred some of the advantages of the Bayesian approach to a constraint-based search method over models with latent confounders. Their BCCD algorithm builds on the skeleton search of PC, and computes Bayesian-style probabilities for the (conditional) independence constraints.

4 WEIGHTING SCHEMES

Given the general search space we are considering, we take a constraint-based approach. But unlike other such methods we do not select tests to perform based on previous test results. Instead, for a given data set, we consider *all* independence tests that can be performed on the set of variables, and apply the following weighting schemes to the resulting constraints.

4.1 Controlling False Negatives

One of the problems for constraint based causal discovery are false negative results, i.e. variables that are truly dependent but test as independent due to low sample sizes or several cancelling d-connecting paths between them (violations of faithfulness). Strictly speaking, classical statistical tests do not license the inference to independence when the null-hypothesis H_0 of independence fails to be rejected. Schulte et al. (2010) have developed a search procedure for causal DAGs based on *dependence* constraints, which are licensed by classical tests when H_0 is rejected. Independencies enter only as a result of a simplicity assumption. Analogously, we propose to control the false negatives with a given sufficiently low p-value threshold on the independence tests. We enforce the detected dependencies as hard constraints. Dependence constraints on their own cannot conflict, as the complete graph will satisfy all possible dependencies. But the principle of Occam’s Razor recommends choosing the simplest among the models able to produce the data. We take this here to amount to maximizing the number of independencies given the dependencies (which is closely related to the dependence minimality proposed by Pearl (2000)). Thus, if we partition the set of constraints \mathbf{K} into the independence constraints \mathbf{K}_\perp and dependence constraints $\mathbf{K}_\mathcal{I}$, then the causal discovery problem over the class of causal models \mathcal{G} amounts to solving the constrained optimization problem in (1) with a weight function

$$w(k) = \begin{cases} \infty & \text{if } k \in \mathbf{K}_\mathcal{I} \\ 1 & \text{if } k \in \mathbf{K}_\perp. \end{cases} \quad (2)$$

⁴Some PC implementations do in some cases infer the presence of a latent confounder (violating the model space) when v-structures are incorrectly detected.

4.2 Controlling False Positives and Negatives

In a slight but common abuse of classical statistics, one can treat the failure to reject H_0 as the acceptance of independence. Then one can simply find the graph that minimizes the number of disagreements between the (in)dependence constraints implied by the graph and the test results. Such an approach would then also be able to recover from false positive errors, i.e. true independencies that test as dependencies. Such false positive errors may occur, for example, if the true structure is a chain $x \rightarrow z \rightarrow y$ and we condition on a measured, but noisy, version of z , and we still get that $x \not\perp y | z$. The corresponding weight-function for the constrained optimization problem in (1) is then simply

$$w(k) = 1 \quad \forall k \in \mathbf{K}. \quad (3)$$

4.3 Weighted Constraints

So far we have treated the test results as binary, but in many cases one has reason to be more confident about some test results than others, and the constraints could be weighted accordingly. One would like to associate a probability as a measure of confidence with each constraint, since then the independence test could be treated as a probabilistic classifier without a hard decision between independence and dependence. When ground truth is available the quality of probabilistic classifiers is often compared by proper scoring rules, such as the log-score. Proper scoring rules assign costs that are minimized when the tests or classifier return the true probability of class membership. We use such scoring rules here as cost functions: we find the graph G^* such that if it were the ground truth, the results of the probabilistic classifier would be optimal – minimizing the cost assigned by the proper score. Given a data set \mathbf{D} , for each constraint k the classifier returns the probability $P(k | \mathbf{D})$ for k to hold in G^* . If in fact k holds in G^* , then the classifier should only suffer the cost $-\log P(k | \mathbf{D})$, otherwise the cost is $-\log[1 - P(k | \mathbf{D})]$. As the minimum of these costs will always be suffered, it is sufficient to let

$$w(k) = \log P(k | \mathbf{D}) - \log[1 - P(k | \mathbf{D})], \quad (4)$$

which is positive, since only the constraint with higher probability is included in \mathbf{K} .⁵

It is not straightforward, neither in terms of theoretical foundation nor actual implementation, to turn p-values from classical tests into probability estimates, as the distribution of p-values under H_0 is uniform and only known to be decreasing under H_1 .⁶ Instead, Margaritis and Bromberg (2009) use a Bayesian paradigm to assign proba-

bilities to the (in)dependence statements.⁷ Following them, for each independence statement $x \perp y | \mathbf{C}$, we consider two models M_{\perp} and $M_{\not\perp}$, where $M_{\perp} : P(x, y | \mathbf{C}) = P(x | \mathbf{C})P(y | \mathbf{C})$ postulates independence, while $M_{\not\perp} : P(x, y | \mathbf{C}) = P(x | \mathbf{C})P(y | x, \mathbf{C})$ postulates dependence. Given data \mathbf{D} and a prior $P(M_{\perp}) = \alpha$ the probability associated with $k = x \perp y | \mathbf{C}$ simplifies to

$$P(k | \mathbf{D}) = \frac{P(y | \mathbf{C})\alpha}{P(y | \mathbf{C})\alpha + P(y | x, \mathbf{C})(1 - \alpha)}.$$

The marginal likelihoods $P(y | \mathbf{C})$ and $P(y | x, \mathbf{C})$ correspond to local scores in the score-based learning framework and have a closed form for categorical variables using a Dirichlet prior (Cooper and Herskovits, 1992) or for continuous variables with linear relations and Gaussian disturbances using an inverse Wishart Gaussian prior (Geiger and Heckerman, 1994).

5 A RECURSIVE VIEW TO CAUSAL GRAPHS

Given a set of (in)dependence constraints, finding an optimal graph G^* (in the sense of the problem statement's objective function Eq. 1) requires a formulation of the d-connection property that is suitable for constraint solvers. Hyttinen et al. (2013) provided such a formulation in terms of propositional logic, but that proves to be inefficient for the computationally more demanding case with conflicted constraints (see Section 7).

Our new formulation is based on the two central graph operations that relate the underlying causal graph to the (in)dependence constraints obtained from an observational data set: *conditioning* and *marginalization*. (Intervening is treated in the supplementary material.) We define these operations over objects that we call *d-connection graphs* rather than MAGs/PAGs (Richardson and Spirtes, 2002), as we want allow for the complete generality of the model space.

A *d-connection graph* $H = (\mathbf{V}, \mathbf{E})_{\mathbf{C}}$ is defined relative to a set \mathbf{C} such that: (i) \mathbf{V} is the set of variables in the graph, (ii) the edge relation $\mathbf{E} = \mathbf{E}_{\rightarrow} \cup \mathbf{E}_{\leftrightarrow} \cup \mathbf{E}_{-}$ is composed of directed, bidirected and undirected edges among \mathbf{V} , and (iii) the set \mathbf{C} denotes the set of conditioned variables with the restriction that $\mathbf{V} \cap \mathbf{C} = \emptyset$. The disjoint sets \mathbf{V} and \mathbf{C} are used to keep track of which variables have been subject to the two operations. Given a causal graph $G = (\mathbf{V}, \mathbf{E})$ (Section 2), the corresponding d-connection graph H has the same sets \mathbf{V} , \mathbf{E}_{\rightarrow} and $\mathbf{E}_{\leftrightarrow}$, but in addition the set $\mathbf{E}_{-} = \emptyset$ (no undirected edges) and $\mathbf{C} = \emptyset$ (no conditioning). The d-connection graph of the causal graph in

⁵In Appendix B we show how the log-weights can be interpreted probabilistically.

⁶Some proposals in this direction appear in a recent unpublished paper by Triantafyllou and Tsamardinos (2014).

⁷Claassen and Heskes (2012) also describe a way of obtaining Bayesian probabilities for independence statements, which is particularly accurate for finding minimal independencies in an acyclic domain.

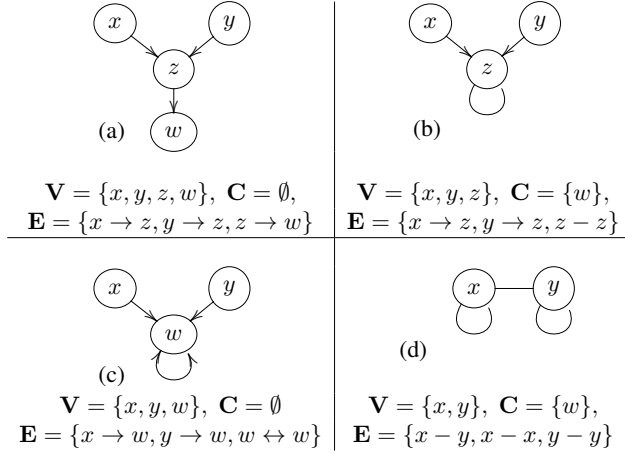


Figure 2: Graph operations on d-connection graphs: (a) original graph, (b) after conditioning on w , (c) after marginalizing z , (d) after conditioning on w and marginalizing z in either order. The xy -edge in (d) shows that x and y are dependent when conditioning on z and marginalizing w .

Figure 1 is shown in Figure 2a; the edges correspond exactly. In general, an edge $x_a - by$ in a d-connection graph $H = (\mathbf{V}', \mathbf{E}')_{\mathbf{C}'}$ denotes the existence of a path p with the given edge ends a and b at x and y in the underlying causal graph G , such that p is d-connecting with respect to \mathbf{C}' and does not go through other variables in \mathbf{V}' . Consequently, the d-connection property (as defined in Section 2) can be directly applied to any d-connection graph.

Given a d-connection graph $H = (\mathbf{V}, \mathbf{E})_{\mathbf{C}}$, the *conditioning* operation $c(H, w)$ on a variable $w \in \mathbf{V}$ results in a d-connection graph $H' = (\mathbf{V} \setminus w, \mathbf{E}')_{\mathbf{C} \cup w}$, where \mathbf{E}' is related to \mathbf{E} by (i) including in \mathbf{E}' any edges in \mathbf{E} not involving w ; (ii) adding to \mathbf{E}' an edge $x_a - by$ if there are edges $x_a \rightarrow w$ and $w \leftarrow by$ in \mathbf{E} , and (iii) not permitting any other edges in \mathbf{E}' (x and y can be equal above).

The graphs in the right column of Figure 2 are formed by applying the operation $c(\cdot, w)$ to the graphs in the left column of Figure 2. The conditioning operation is a little unusual because the conditioned variable is removed from the graph and undirected edges and undirected self-cycles are introduced. As is well-known, conditioning on a collider or a child of a collider results in a d-connection between the collider’s parents. For example, if w is conditioned on in $x \rightarrow w \leftarrow y$, we have a d-connecting path between x and y with a tail at either end. Since the encoding also removes the conditioned variable from the graph, we just have an undirected edge $x - y$, the parents are “moralized”. The undirected self-loop represents the same idea, just in case of the child of a collider: each parent of a conditioned variable receives an undirected self-loop to indicate that it can provide a d-connection between two incoming paths. In Figure 2a-b this is illustrated for variable z when w is added

to the conditioning set (and removed from the graph). The graph in (b) indicates that there is a tail to tail d-connection at z , which implies that x and y are now d-connected when w , the child of collider z , is in the conditioning set. The (perhaps unintuitive) removal of the conditioning variable from the graph achieves two goals: it reduces the size of the graph to be encoded, but more importantly, it incrementally represents the effect of conditioning on the d-connections among the other variables.

Given a d-connection graph $H = (\mathbf{V}, \mathbf{E})_{\mathbf{C}}$, the *marginalization* operation $m(H, z)$ for variable $z \in \mathbf{V}$ results in a d-connection graph $H' = (\mathbf{V} \setminus z, \mathbf{E}')_{\mathbf{C}}$, where \mathbf{E}' is related to \mathbf{E} by (i) including in \mathbf{E}' any edges in \mathbf{E} not involving z ; (ii) adding to \mathbf{E}' an edge $x_a - by$ if there are edges $x_a \rightarrow z$ and $z \leftarrow by$ in \mathbf{E} ; (iii) adding to \mathbf{E}' an $x_a - by$ if there are edges $x_a \rightarrow z$, $z - z$ and $z \leftarrow by$ in \mathbf{E} , and (iv) not permitting any other edges in \mathbf{E}' (x and y can be equal above). Marginalization follows the standard graphical procedures used elsewhere, except that a little more book-keeping is required to track the d-connections resulting from self-loops and undirected edges (see Figure 2 top to bottom).

The following theorem shows that the operations preserve the d-connection properties among the variables still present in the graph after the operation (proof in Appendix D).

Theorem 1 *Let $H' = (\mathbf{V}', \mathbf{E}')_{\mathbf{C}'}$ be the d-connection graph obtained from a d-connection graph $H = (\mathbf{V}, \mathbf{E})_{\mathbf{C}}$ by applying the conditioning operation on w , i.e. $H' = c(H, w)$ (or by applying the marginalization operation on z , thus $H' = m(H, z)$). Then there is a path of type $x_a \dots by$ that is d-connecting given $\mathbf{C}'' \supseteq \mathbf{C}'$ in H' if and only if there is a path of type $x_a \dots by$ that is d-connecting given \mathbf{C}'' in H .*

Consequently, a dependence $x \not\perp\!\!\!\perp y \mid \mathbf{C}$ in (the true causal graph) $G = (\mathbf{V}, \mathbf{E})$ is equivalent to having an edge of some type in the d-connection graph $H = (\mathbf{V}', \mathbf{E}')_{\mathbf{C}}$ when $\mathbf{V}' = \{x, y\}$. This d-connection graph H can be obtained by consecutively applying the conditioning operation to all variables in \mathbf{C} , and the marginalization operation to the rest in $\mathbf{V} \setminus (\mathbf{C} \cup \{x, y\})$. For example, in Figure 2d the undirected edge between x and y represents the fact that $x \not\perp\!\!\!\perp y \mid w$ in the underlying causal graph (Figure 1).

Given a causal graph G , we can calculate all of its implied (in)dependence relations by applying the operations in any order. However, we need not apply all operations for each (in)dependence relation, since we can exploit the compact intermediary representations of the d-connections in the d-connection graphs obtained earlier. Figure 3 shows the “encoding DAG” for one set of applied operations by which we obtain all (in)dependence relations of the four variable graph in Figure 1. The true causal graph corresponds to the node in the middle, and we use the operations to move

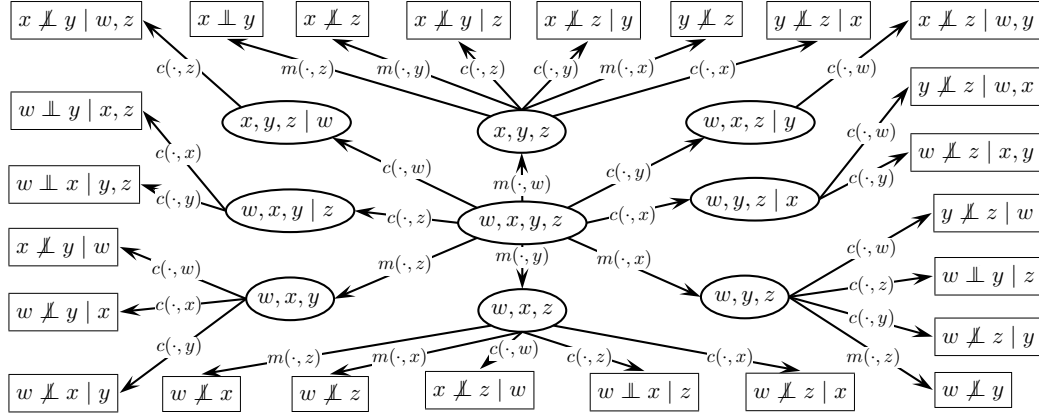


Figure 3: An “encoding DAG”: a tree connecting the causal graph (represented by the middle node) to (in)dependence constraints (leaves). The conditioning/marginalization operations are marked on the arrows. The circular nodes represent the sets $\mathbf{V} \mid \mathbf{C}$ of the corresponding d-connection graphs and the square nodes show the implied independencies for the true causal graph shown in Figure 1.

outward until we reach the implied (in)dependencies at the leaves.

For causal discovery we want to perform the backward inference. We know the (in)dependence relations at the leaves (Figure 3) and our aim is to find a causal graph in the middle node. Again we exploit the same tree structure. But this time we will have at each node several different graphs that satisfy the constraints, which are downstream from it. Moving towards the center, nodes can be combined to find the options that satisfy all constraints downstream from the node. For example, in the node ‘ x, y, z ’ all the tests relevant to identifying the v-structure $x_a \rightarrow z \leftarrow b y$ are available locally. In the general case of weighted and possibly conflicting (in)dependence constraints, the inference becomes hard. We use off-the-shelf solvers for this job. Nevertheless, as shown by the simulations, the recursive structure allows the backwards inference to work faster than for the logical formulations of d-connection in Hyttinen et al. (2013).

6 CAUSAL DISCOVERY VIA ASP

Building on Section 5, we describe here an ASP-based constraint optimization approach to optimally solving the causal structure discovery task defined in (1).

Answer set programming (ASP) is a rule-based declarative constraint satisfaction paradigm that is well-suited for representing and solving various computationally hard problems (Gelfond and Lifschitz, 1988; Niemelä, 1999; Simons et al., 2002). ASP offers an expressive declarative modelling language in terms of first-order logical rules, allowing for intuitive and compact representations of NP-hard optimization tasks. When using ASP, the first task is to model the problem in terms of ASP rules (constraints) so

that the set of solutions implicitly represented by the ASP rules corresponds to the solutions of the original problem. One or multiple solutions of the original problem can then be obtained by invoking an off-the-shelf ASP solver on the constraint declaration.

6.1 An ASP Encoding of Causal Discovery

As a self-contained explanation of ASP syntax and semantics would exceed the page limit, we only aim to give an intuitive reading of our ASP encoding. The ASP encoding, outlined in Figure 4, is based on exactly representing the conditioning and marginalization operations (defined in Section 5) in ASP.

Answer set programming can be viewed as a data-centric constraint satisfaction paradigm, in which the input data, represented as “facts” that are true via input predicates, express the instance of the original problem at hand. In our case, the problem instance consists of a set of independence and dependence constraints and their associated weights, represented via the predicates *indep* and *dep*, and the “encoding DAG”, describing which d-connection graphs can be mapped from one to the other via the conditioning and marginalization operations (predicates *cond* and *marg*). Concretely, the input predicates *indep*($x, y, \{x, y\}, \mathbf{C}, W$) and *dep*($x, y, \{x, y\}, \mathbf{C}, W$) represent as facts that the input contains an independence (resp., dependence) constraint with weight W over the variables x and y given the conditioning set \mathbf{C} .⁸ The input predicates *cond*($\mathbf{V}, \mathbf{C}, z$) and *marg*($\mathbf{V}, \mathbf{C}, z$) enable the conditioning and marginalizing of a variable z , respectively, in a d-connection graph that has exactly the variables \mathbf{V} and conditioning set \mathbf{C} . Es-

⁸In practice, the ASP language requires integer-valued weights. For sufficient precision, in our experiments we multiply the weights by 1000, and then truncate to integers.

sentially, *cond* and *marg* represent edges in the encoding DAG (recall Figure 3).

The other predicates $tt(x, y, \mathbf{V}, \mathbf{C})$, $th(x, y, \mathbf{V}, \mathbf{C})$, and $hh(x, y, \mathbf{V}, \mathbf{C})$ present the existence of different types of edges (tt: tail-tail, th: tail-head, hh: head-head) in

Conditioning on a variable $z \in \mathbf{V}, \forall x, y \in \mathbf{V} \setminus z$:

$$\begin{aligned} th(x, y, \mathbf{V} \setminus z, \mathbf{C} \cup z) &:- th(x, y, \mathbf{V}, \mathbf{C}), cond(\mathbf{V}, \mathbf{C}, z). \\ th(x, y, \mathbf{V} \setminus z, \mathbf{C} \cup z) &:- th(x, z, \mathbf{V}, \mathbf{C}), hh(z, y, \mathbf{V}, \mathbf{C}), \\ &\quad cond(\mathbf{V}, \mathbf{C}, z). \\ hh(x, y, \mathbf{V} \setminus z, \mathbf{C} \cup z) &:- hh(x, y, \mathbf{V}, \mathbf{C}), cond(\mathbf{V}, \mathbf{C}, z). \\ hh(x, y, \mathbf{V} \setminus z, \mathbf{C} \cup z) &:- hh(x, z, \mathbf{V}, \mathbf{C}), hh(z, y, \mathbf{V}, \mathbf{C}), \\ &\quad cond(\mathbf{V}, \mathbf{C}, z). \\ tt(x, y, \mathbf{V} \setminus z, \mathbf{C} \cup z) &:- tt(x, y, \mathbf{V}, \mathbf{C}), cond(\mathbf{V}, \mathbf{C}, z). \\ tt(x, y, \mathbf{V} \setminus z, \mathbf{C} \cup z) &:- th(x, z, \mathbf{V}, \mathbf{C}), th(y, z, \mathbf{V}, \mathbf{C}), \\ &\quad cond(\mathbf{V}, \mathbf{C}, z). \end{aligned}$$

Marginalizing a variable $z \in \mathbf{V}, \forall x, y \in \mathbf{V} \setminus z$:

$$\begin{aligned} th(x, y, \mathbf{V} \setminus z, \mathbf{C}) &:- th(x, y, \mathbf{V}, \mathbf{C}), marg(\mathbf{V}, \mathbf{C}, z). \\ th(x, y, \mathbf{V} \setminus z, \mathbf{C}) &:- tt(x, z, \mathbf{V}, \mathbf{C}), th(z, y, \mathbf{V}, \mathbf{C}), \\ &\quad marg(\mathbf{V}, \mathbf{C}, z). \\ th(x, y, \mathbf{V} \setminus z, \mathbf{C}) &:- th(x, z, \mathbf{V}, \mathbf{C}), th(z, y, \mathbf{V}, \mathbf{C}), \\ &\quad marg(\mathbf{V}, \mathbf{C}, z). \\ th(x, y, \mathbf{V} \setminus z, \mathbf{C}) &:- tt(x, z, \mathbf{V}, \mathbf{C}), hh(z, y, \mathbf{V}, \mathbf{C}), \\ &\quad marg(\mathbf{V}, \mathbf{C}, z). \\ th(x, y, \mathbf{V} \setminus z, \mathbf{C}) &:- th(x, z, \mathbf{V}, \mathbf{C}), tt(z, z, \mathbf{V}, \mathbf{C}), \\ &\quad hh(z, y, \mathbf{V}, \mathbf{C}), marg(\mathbf{V}, \mathbf{C}, z). \\ hh(x, y, \mathbf{V} \setminus z, \mathbf{C}) &:- hh(x, y, \mathbf{V}, \mathbf{C}), marg(\mathbf{V}, \mathbf{C}, z). \\ hh(x, y, \mathbf{V} \setminus z, \mathbf{C}) &:- th(z, x, \mathbf{V}, \mathbf{C}), th(z, y, \mathbf{V}, \mathbf{C}), \\ &\quad marg(\mathbf{V}, \mathbf{C}, z). \\ hh(x, y, \mathbf{V} \setminus z, \mathbf{C}) &:- hh(x, z, \mathbf{V}, \mathbf{C}), th(z, y, \mathbf{V}, \mathbf{C}), \\ &\quad marg(\mathbf{V}, \mathbf{C}, z). \\ hh(x, y, \mathbf{V} \setminus z, \mathbf{C}) &:- th(z, x, \mathbf{V}, \mathbf{C}), hh(z, y, \mathbf{V}, \mathbf{C}), \\ &\quad marg(\mathbf{V}, \mathbf{C}, z). \\ hh(x, y, \mathbf{V} \setminus z, \mathbf{C}) &:- hh(x, z, \mathbf{V}, \mathbf{C}), tt(z, z, \mathbf{V}, \mathbf{C}), \\ &\quad hh(z, y, \mathbf{V}, \mathbf{C}), marg(\mathbf{V}, \mathbf{C}, z). \\ tt(x, y, \mathbf{V} \setminus z, \mathbf{C}) &:- tt(x, y, \mathbf{V}, \mathbf{C}), marg(\mathbf{V}, \mathbf{C}, z). \\ tt(x, y, \mathbf{V} \setminus z, \mathbf{C}) &:- tt(x, z, \mathbf{V}, \mathbf{C}), tt(z, y, \mathbf{V}, \mathbf{C}), \\ &\quad marg(\mathbf{V}, \mathbf{C}, z). \\ tt(x, y, \mathbf{V} \setminus z, \mathbf{C}) &:- th(x, z, \mathbf{V}, \mathbf{C}), tt(z, y, \mathbf{V}, \mathbf{C}), \\ &\quad marg(\mathbf{V}, \mathbf{C}, z). \\ tt(x, y, \mathbf{V} \setminus z, \mathbf{C}) &:- tt(x, z, \mathbf{V}, \mathbf{C}), th(y, z, \mathbf{V}, \mathbf{C}), \\ &\quad marg(\mathbf{V}, \mathbf{C}, z). \\ tt(x, y, \mathbf{V} \setminus z, \mathbf{C}) &:- th(x, z, \mathbf{V}, \mathbf{C}), tt(z, z, \mathbf{V}, \mathbf{C}), \\ &\quad th(y, z, \mathbf{V}, \mathbf{C}), marg(\mathbf{V}, \mathbf{C}, z). \end{aligned}$$

Inferring failures to sat. (in)dep. $\forall x \forall y > x, \forall \mathbf{C}, \mathbf{V} = \{x, y\}$:

$$\begin{aligned} fail(x, y, \mathbf{V}, \mathbf{C}, W) &:- tt(x, y, \mathbf{V}, \mathbf{C}), indep(x, y, \mathbf{V}, \mathbf{C}, W). \\ fail(x, y, \mathbf{V}, \mathbf{C}, W) &:- th(x, y, \mathbf{V}, \mathbf{C}), indep(x, y, \mathbf{V}, \mathbf{C}, W). \\ fail(x, y, \mathbf{V}, \mathbf{C}, W) &:- th(y, x, \mathbf{V}, \mathbf{C}), indep(x, y, \mathbf{V}, \mathbf{C}, W). \\ fail(x, y, \mathbf{V}, \mathbf{C}, W) &:- hh(x, y, \mathbf{V}, \mathbf{C}), indep(x, y, \mathbf{V}, \mathbf{C}, W). \\ fail(x, y, \mathbf{V}, \mathbf{C}, W) &:- not th(x, y, \mathbf{V}, \mathbf{C}), not th(y, x, \mathbf{V}, \mathbf{C}), \\ &\quad not hh(x, y, \mathbf{V}, \mathbf{C}), not tt(x, y, \mathbf{V}, \mathbf{C}), \\ &\quad dep(x, y, \mathbf{V}, \mathbf{C}, W). \end{aligned}$$

Weak constraints $\forall x \forall y > x, \forall \mathbf{C}, \mathbf{V} = \{x, y\}$:

$$\sim fail(x, y, \mathbf{V}, \mathbf{C}, W). [W]$$

Figure 4: The ASP encoding.

the d-connection graph with variable set \mathbf{V} and conditioning set \mathbf{C} . The rules associated with the conditioning and marginalization operations encode how the different edges in the d-connection graphs are derived from edges in other d-connection graphs through the conditioning and marginalization operations. The restrictions that the (in)dependence constraints put on the set of solutions (causal graphs) follow from these rules. As an example, the first marginalization rule

$$th(x, y, \mathbf{V} \setminus z, \mathbf{C}) :- th(x, y, \mathbf{V}, \mathbf{C}), marg(\mathbf{V}, \mathbf{C}, z).$$

allows to derive, for any choice of $\mathbf{C}, \mathbf{V}, z \in \mathbf{V}$, and $x, y \in \mathbf{V} \setminus z$, that $th(x, y, \mathbf{V} \setminus z, \mathbf{C})$ is true (i.e., that there is an edge $x \rightarrow y$ in the d-connection graph over $\mathbf{V} \setminus z$ relative to \mathbf{C}) given that (i) $th(x, y, \mathbf{V}, \mathbf{C})$ is true (there is an edge $x \rightarrow y$ in the d-connection graph over \mathbf{V} relative to \mathbf{C}), and (ii) the input contains the fact $marg(\mathbf{V}, \mathbf{C}, z)$, i.e., marginalizing z in the d-connection graph over \mathbf{V} relative to \mathbf{C} is allowed by the encoding DAG. Note that the set of derivation rules for $tt(x, y, \mathbf{V}, \mathbf{C})$, $th(x, y, \mathbf{V}, \mathbf{C})$, and $hh(x, y, \mathbf{V}, \mathbf{C})$ are very similar to each other.

Finally, the objective function under minimization (Equation 1) is expressed using the so-called *weak constraints* offered by the ASP language. First, the predicate *fail* is derived whenever the candidate solution disagrees with the input. The first four rules for *fail* denote cases where the constraints tested from the data suggest independence but an edge *tt/th/hh* is derived. The fifth rule derives *fail* whenever the input constraints suggest dependence but there are no d-connecting paths. The last rule of the encoding denotes the fact that whenever the *fail* predicate is derived, then the cost W is incurred. This implies that any solution produced by an ASP solver on the encoding is guaranteed to present an optimal solution to the causal discovery task, as stated by the following theorem.

Theorem 2 *Given a set \mathbf{K} of conditional independence and dependence constraints over a set \mathbf{V} of variables, and a non-negative weight $w(k)$ for each $k \in \mathbf{K}$, for any encoding DAG connecting the (in)dependence constraints, it holds that any optimal solution (minimizing the sum of the unsatisfied weak constraints) to the ASP encoding corresponds to a causal graph that minimizes the objective function Equation 1.*

To find optimal solutions to the encoded problem, the input facts together with the derivation rules and weak constraints are given as input to an ASP solver. The actual complete search for solutions is then performed over a propositional instantiation of the first-order rules; this process is automatized within the ASP solver and does not require involvement of the user. The search procedure implemented within state-of-the-art ASP solvers, such as Clingo (Gebser et al., 2011) used in this paper, is then based on the successful and highly-efficient Boolean satisfiability solver technology (Biere et al., 2009).

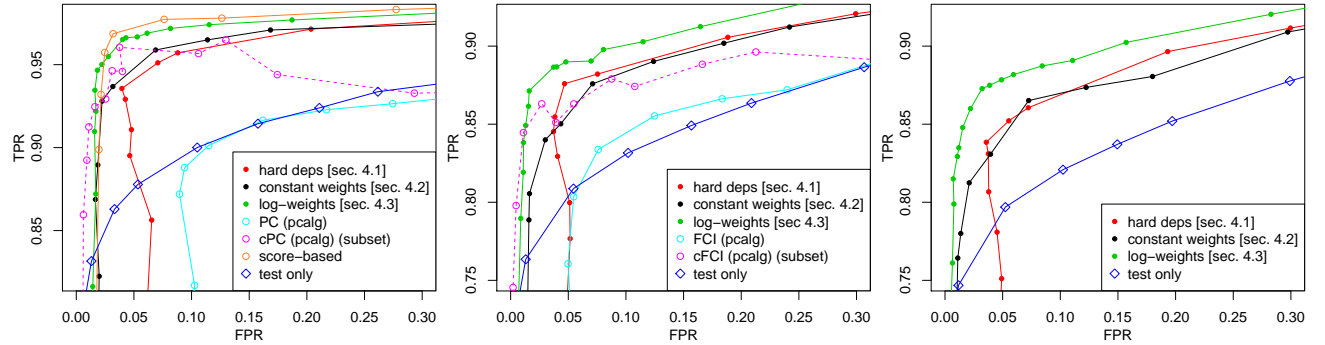


Figure 5: ROC given causally sufficient acyclic models (left), causally insufficient acyclic models (middle) and causally insufficient cyclic models (right).

7 SIMULATIONS

We first test the accuracy of our method against several competing algorithms under the restricting assumption of acyclicity (and causal sufficiency), since no other algorithms are presently available for the most general model space that we can handle. In the tests for accuracy we draw 200 linear Gaussian models over 6 variables, where the edge coefficients are drawn uniformly from $\pm[0.2, 0.8]$. The directed edges were drawn randomly such that the average degree of each node was 2 for the causally sufficient models. For models with latent confounders the average degree of the nodes for directed edges was 1, and the covariance matrix of the disturbances corresponded to the passively observed covariance matrix of another similar causally sufficient linear Gaussian model. For all methods we use a correlation based t-test, or the corresponding Bayesian test. Although our methods straightforwardly allow for experimental data sets, we only used 500 samples of passively observed data here to adhere to the restrictions of the competing methods.

Different methods represent the uncertainty in their output differently: cPC and cFCI return parts of the graphs as unknown, while score-based methods, as well as the approaches in this paper, may be used to return several high scoring graphs. We compared the methods regarding only the single ‘highest scoring’ graph (or Markov equivalence class). To account for the fact that each method returns an equivalence class based on its model space assumptions, we evaluate the d-separation and d-connection relations of the learned result against those of the true data generating graph. This includes all possible d-connection/separation relations in the passively observed setting. Each of the methods takes a parameter (such as a p-value-threshold) that adjusts the sparsity of the output. To avoid effects of a specific parameter choice, we plot the accuracy of the different methods run with different parameters in the ROC space.

Figure 5 (left) shows the accuracy of the methods for acyclic and causally sufficient data generating models. The

blue line shows the performance of the tests alone (classic t-tests almost exactly equal the Bayesian tests in the ROC space). The plain PC algorithm (implemented by Kalisch et al. (2012)) does not exceed the performance of these tests and cannot achieve high true positive rates for acceptable false positive rates. cPC returns an equivalence class of graphs without unknown parts in only 58/200 cases for the optimal p-value threshold of 0.1 (only outputs from these runs are considered in the plot for cPC). Its performance on these ‘easy instances’ is quite good. The score-based approach achieves much better accuracy (BIC score; the MAP DAG is found using exact methods). All our approaches were run by restricting the model space to acyclic and causally sufficient graphs. The surprising finding here is that the constraint-based approach using ‘log-weights’, introduced in Section 4.3, seems to be able to roughly match the performance of the score-based method. This suggests that the constraint-optimization resolves many conflicts that arise from erroneous tests, and so it is an accurate approach for causal discovery. Also, the other suggested approaches, ‘hard deps’ (Section 4.1) and ‘constant weights’ (Section 4.2), seem to be able to perform quite well. Clearly, their results are not restricted to that of the test performance. The conflict resolution is able to correct many erroneous test results.

Figure 5 (middle) shows the accuracy of the methods in the ROC-space assuming acyclicity but not causal sufficiency. Our approaches (now only restricted by acyclicity), especially the ‘log-weights’, achieve higher true positive rates than the competing methods. Again cFCI does better than FCI but only returns a fully determined result for 61/200 of the cases for the optimal p-value threshold of 0.1. For these ‘easier instances’ (again only results without unknowns are plotted for cFCI) its performance is quite good.⁹ Figure 5 (right) shows the accuracy of the proposed methods in the

⁹The BCCD algorithm (Claassen and Heskes, 2012) and the (still unpublished) approach of Triantafyllou and Tsamardinos (2014) would provide the most suitable comparison in this setting. We hope to perform the comparison when implementations of the methods are made available.

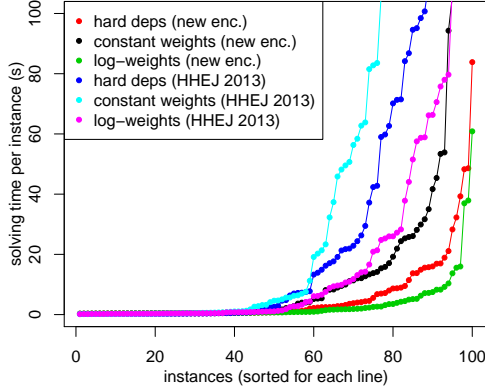


Figure 6: Solving times for (possibly) cyclic causally insufficient models (right).

most general model space allowing for cycles and latent confounders. There currently do not exist other methods that apply to such a general model space.

Figure 6 shows the solving times of `Clingo` on a 2.4-GHz Intel Core i5 processor for 100 instances of constraints obtained from 500 samples of passively observed data generated by different possibly cyclic and causally insufficient models over six variables. The solving times are sorted for each algorithm individually. The plot shows nicely the rather large variance of the solving times. Easy problems (left on the plot) are solved almost instantly, while harder problems may take considerably longer. This is a general feature of exact algorithms solving very complex problems: in the worst case the NP-complexity of the problem kicks in, but still a large number of instances are relatively easy to solve. In the figure we also compare the present encoding against a straightforward ASP-implementation of the encoding of Hyttinen et al. (2013). For all weighting schemes, and especially for the harder instances, the encoding presented in this paper seems to allow for much faster solving. Different weighting schemes also clearly affect the solving time. ‘log-weights’ and ‘hard deps’ are considerably faster than ‘constant weights’. For graphs with seven observed variable the solving times take up to half an hour (see supplement), for graphs with eight variables many instances take several hours to solve.

Finally we analyzed what actually happens in the conflict resolution. We generated data from 100 random parameterizations of the graph in Figure 1, and ran the inference with log-weights for different sample sizes (Figure 7). The redness of the background color denotes how many times the specific independence tests most prone to error (listed on the right axis) produced an incorrect result. The blue line counts the number of incorrect tests obtained from the data, which serve as input to our method. The black line shows the number of (in)dependence relations that were incorrect in the output graph of our method. The tests produce errors

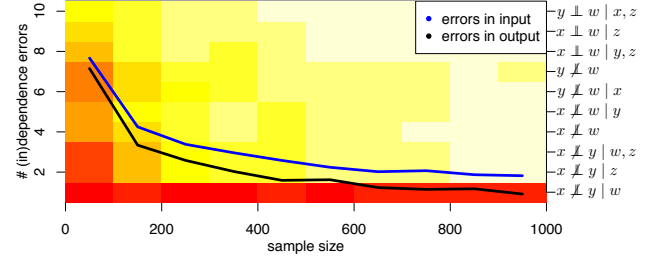


Figure 7: Conflict resolution for the graph in Figure 1.

throughout different sample sizes, but we are able to offer an output with significantly fewer errors.

8 CONCLUSION

We presented a method for causal discovery that works in a completely general search space that includes models with feedback *and* latent confounders. It accepts as input possibly inconsistent (in)dependence constraints obtained from overlapping experimental or observational data sets. It returns an *exact* output, in the sense that it finds the graph that minimizes a well-defined objective function, which characterizes how conflicted constraints should be resolved. We have considered a variety of theoretical motivations for different conflict resolution schemes, and tested them successfully against several extant algorithms.¹⁰ Although not shown explicitly in this article, the direct encoding of the d-connection properties ensures that in the infinite sample limit, our algorithm retains the completeness properties of Hyttinen et al. (2013).

The scalability of the present procedure is still quite limited, but it is similar to that of the exact graph structure discovery methods exploiting ASP of Corander et al. (2013), who search for (undirected) Markov networks. We have emphasized the exactness of our method, which provides theoretical guarantees and a very high accuracy. All other methods we are aware of that consider similar search spaces take a greedy approach in one way or another. Our results suggest that the resulting scalability may come at the price of accuracy for realistic sample sizes on anything but very sparse causal structures. We hope to use our approach as basis to explore the trade-off between scalability and accuracy in the future.

Acknowledgements

A.H. was supported by the Finnish Foundation for Technology Promotion (TES), and M. J. by the Academy of Finland under grants 251170 (COIN Finnish Centre of Excellence in Computational Inference Research) and 276412.

¹⁰The supplementary material and the code package reproducing the simulation results is available from the authors’ websites.

References

- Biere, A., Heule, M., van Maaren, H., and Walsh, T., editors (2009). *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*. IOS Press.
- Chickering, D. M. (2002). Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3:507–554.
- Claassen, T. and Heskes, T. (2012). A bayesian approach to constraint based causal inference. In *Proceedings of UAI*, pages 207–216. AUAI Press.
- Cooper, G. F. and Herskovits, E. (1992). A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347.
- Corander, J., Janhunen, T., Rintanen, J., Nyman, H. J., and Pensar, J. (2013). Learning chordal Markov networks by constraint satisfaction. In *Proceedings of NIPS*, pages 1349–1357.
- Gebser, M., Kaufmann, B., Kaminski, R., Ostrowski, M., Schaub, T., and Schneider, M. T. (2011). Potassco: The Potsdam answer set solving collection. *AI Communications*, 24(2):107–124.
- Geiger, D. and Heckerman, D. (1994). Learning Gaussian networks. Technical Report MSR-TR-94-10, Microsoft Research.
- Gelfond, M. and Lifschitz, V. (1988). The stable model semantics for logic programming. In *Logic Programming, Proceedings of the Fifth International Conference and Symposium*, pages 1070–1080.
- Hytinen, A., Hoyer, P. O., Eberhardt, F., and Järvisalo, M. (2013). Discovering cyclic causal models with latent variables: A general SAT-based procedure. In *Proceedings of UAI*, pages 301–310. AUAI Press.
- Kalisch, M., Mächler, M., Colombo, D., Maathuis, M. H., and Bühlmann, P. (2012). Causal inference using graphical models with the R package pcalg. *Journal of Statistical Software*, 47(11):1–26.
- Koster, J. T. A. (2002). Marginalizing and conditioning in graphical models. *Bernoulli*, 8(6):817–840.
- Li, J. and Wang, Z. J. (2009). Controlling the false discovery rate of the association/causality structure learned with the PC algorithm. *Journal of Machine Learning Research*, 10:475–514.
- Margaritis, D. and Bromberg, F. (2009). Efficient Markov network discovery using particle filters. *Computational Intelligence*, 25(4):367–394.
- Neal, R. (2000). On deducing conditional independence from d-separation in causal graphs with feedback. *Journal of Artificial Intelligence Research*, 12:87–91.
- Niemelä, I. (1999). Logic programs with stable model semantics as a constraint programming paradigm. *Annals of Mathematics and Artificial Intelligence*, 25(3-4):241–273.
- Pearl, J. (2000). *Causality: Models, Reasoning, and Inference*. Cambridge University Press.
- Pearl, J. and Dechter, R. (1996). Identifying independencies in causal graphs with feedback. In *Proceedings of UAI*, pages 420–426. Morgan Kaufmann.
- Ramsey, J., Zhang, J., and Spirtes, P. (2006). Adjacency-faithfulness and conservative causal inference. In *Proceedings of UAI*, pages 401–408. AUAI Press.
- Richardson, T. and Spirtes, P. (2002). Ancestral graph Markov models. *Annals of Statistics*, 30(4).
- Schulte, O., Luo, W., and Greiner, R. (2010). Mind change optimal learning of Bayes net structure from dependency and independency data. *Information and Computation*, 208(1):63 – 82.
- Simons, P., Niemelä, I., and Soininen, T. (2002). Extending and implementing the stable model semantics. *Artificial Intelligence*, 138(1-2):181–234.
- Spirtes, P. (1995). Directed cyclic graphical representation of feedback models. In *Proceedings of UAI*, pages 491–498. Morgan Kaufmann.
- Spirtes, P., Glymour, C., and Scheines, R. (1993). *Causation, Prediction, and Search*. Springer-Verlag.
- Studený, M. (1998). Bayesian networks from the point of view of chain graphs. In *Proceedings of UAI*, pages 496–503. Morgan Kaufmann.
- Triantafillou, S. and Tsamardinos, I. (2014). Constraint-based causal discovery from multiple interventions over overlapping variable sets. arXiv:1403.2150 (unpublished).
- Triantafillou, S., Tsamardinos, I., and Tollis, I. G. (2010). Learning causal structure from overlapping variable sets. In *Proceedings of AISTATS*, pages 860–867. JMLR.
- Tsamardinos, I., Triantafillou, S., and Lagani, V. (2012). Towards integrative causal analysis of heterogeneous data sets and studies. *Journal of Machine Learning Research*, 13:1097–1157.

Generating structure of latent variable models for nested data

Masakazu Ishihata

NTT Communication Science Laboratories
ishihata.masakazu@lab.ntt.co.jp

Tomoharu Iwata

NTT Communication Science Laboratories
iwata.tomoharu@lab.ntt.co.jp

Abstract

Probabilistic latent variable models have been successfully used to capture intrinsic characteristics of various data. However, it is nontrivial to design appropriate models for given data because it requires both machine learning and domain-specific knowledge. In this paper, we focus on data with nested structure and propose a method to automatically generate a latent variable model for the given nested data, with the proposed method, the model structure is adjustable by its structural parameters. Our model can represent a wide class of hierarchical and sequential latent variable models including mixture models, latent Dirichlet allocation, hidden Markov models and their combinations in multiple layers of the hierarchy. Even when deeply-nested data are given, where designing a proper model is difficult even for experts, our method generate an appropriate model by extracting the essential information. We present an efficient variational inference method for our model based on dynamic programming on the given data structure. We experimentally show that our method generates correct models from artificial datasets and demonstrate that models generated by our method can extract hidden structures of blog and news article datasets.

1 INTRODUCTION

Probabilistic latent variable models have been successfully used for analyzing and capturing intrinsic characteristics of a wide variety of datasets. They are applied to various tasks such as dimension reduction, clustering, visualization and cluster matching [14, 28, 16, 15]. However, designing appropriate models for given data is difficult because it requires machine learning knowledge to formulate models and derive algorithms, and also domain-specific knowledge

to introduce appropriate latent variables and their dependencies.

In this paper, we focus on data with a nested structure and aim to automatically generate a latent variable model that is appropriate to the given nested data. Many data have nested structures such as hierarchies. A document contains sentences and each of the sentences contains words. Purchase data consist of purchase histories of many users, where a user history contains multiple shopping events and a shopping event is represented as a basket containing items that are purchased at the same time. Also music shows such hierarchy; many musicians compose music scores by combining multiple phrases, where each of the phrases consists of musical notes. Such nested data sometimes contain not only hierarchical information but also sequential information, that is, the order of elements in nested groups such as word order in a sentence. Even though such rich structural information is given, we do not need to incorporate all of the information in models. For example, a latent Dirichlet allocation (LDA) model [7], which is a widely-used Bayesian latent variable model for text data, assumes a document is a bag of words; it ignores word order. In contrast, a hidden Markov model (HMM), which is the most famous latent variable model for sequential data, assumes a document is a sequence of words but ignores its hierarchical information. When we design models for such structured data, we have to extract essential information and reflect that into models by introducing several modeling assumptions.

The main contribution of this paper is threefold. First, we propose a latent variable model based on a hierarchical and sequential structure of the given data, where our model can adjust its model structure by *structural parameters*. By changing the values of structural parameters, it can represent various models including mixture models, LDA models, HMMs and their combinations. Second, we propose a universal variational inference algorithm for our model based on dynamic programming on the given data structure. Even if model structures and data structures are changed, we can efficiently compute variational free energy

(VFE), which is used for a model scoring function [5], by our universal algorithm. Third, we formulate the model generation task as an optimization problem for maximizing VFE with respect to structural parameters. No matter how deep the input nested data are, our method extracts the essential information by adjusting model structure. Our method can generate complex models if necessary. Otherwise, it generates simple models.

The remainder of this paper is organized as follows. We first briefly review related work in Section 2. We then introduce an ordered tree representation for nested data in Section 3. In Section 4 we propose our latent variable model and our model generation method. We describe its efficient variational inference algorithm in Section 5. In Section 6, we illustrate our method by experiments using artificial and real-world datasets. Finally, we summarize this paper in Section 7.

2 RELATED WORK

A number of methods for automatically generating model structures from data have been proposed. Structure learning for Bayesian networks (BNs) [22] is one such example. A BN defines a joint distribution over random variables by a directed acyclic graph (DAG) and model parameters. A DAG defines conditional independencies over random variables and model parameters define their conditional distributions. BN structure learning (BNSL) [17] is a problem to find the DAG that maximizes a scoring function from a model class. Unfortunately, the problem is generally NP-hard [8, 9]. Most existing BNSL methods assume no latent variable or limit the number of latent variables [5, 24, 26, 27, 20]. Whereas our aim is to obtain appropriate latent variable models for the given data, the aim of BNSL is to obtain the exact dependencies of observable variables. It is intractable to naively apply existing BNSL methods to our aim because introducing latent variables exponentially increases the number of candidate model structures and the computation time of a scoring function. Our method avoids such intractability by restricting our model class based on the given nested structure and by introducing a universal algorithm for efficient score computation on the restricted model class.

A hierarchical latent class (HLC) model [28] is a tree-structured BN with latent variables, where its leaf nodes are observable variables and the others are latent variables. Learning HLC model structure is similar to our aim, however, it differs in the following two points. First, our method assumes that data have nested structure and exploits the data structure to generate appropriate models. Second, our method considers both hierarchical and sequential dependencies of latent variables. Model structure generated by our method includes, but are not limited to, tree structures.

A model generation method based on matrix factorization has been proposed [12]. The method defines its model class by using context-free grammar of which grammatical rules correspond to matrix factorization processes. A sentence generated by the grammar represents a hierarchical matrix factorization model. The method assumes that given data are represented as a single matrix whereas our method uses data together with its nested structure.

Several latent variable models for nested data have been proposed. The segmented topic model (STM) [11], which is a variant of probabilistic topic models [6], assumes that a document is a collection of segments and each of the segments is a collection of words. The STM captures correlations of topics over segments and segments over documents by introducing segment-level and document-level topic probabilities. The tree-informed LDA (tiLDA) [18], which is a multi-level LDA model, exploits another type of hierarchical structure of documents. The tiLDA model assumes that documents are grouped into categories and each of the categories are also grouped into another categories. Those models exploit inner-document and outer-document nested structures, respectively. Our model also copes with both inner and outer document hierarchies. It additionally exploits sequential information such as the word and segment order. By automatically extracting the essential part of such hierarchical and sequential structures, it generates appropriate models for the given nested data.

Similarly, latent variable models for sequential data have been proposed. A hidden Markov topic model (HMTM) [3], which is a natural extension of LDA models for the nonexchangeable setting, assumes that word topics are generated by a Markov chain. Each document has its own topic Markov chain, however, word emission probabilities of topics are shared by all documents. A hidden topic Markov model (HTMM) [13] introduces a Markov chain to LDA models with a different fashion from HMTM. The HTMM assumes that words in the same sentence share a common topic and each of the sentence topics is generated by a Markov chain. Similar to those methods, our method can combine LDA models and HMMs. Additionally, our method automatically chooses appropriate levels in which Markov chains should be introduced.

3 ORDERED TREE FOR NESTED DATA

We here introduce an ordered tree representation of nested data. Suppose that nested data \mathbf{D} is represented as a pair (\mathbf{x}, \mathbf{T}) , where $\mathbf{x} \equiv (x_n)_{n=1}^N$ is a sequence of observable variables and \mathbf{T} is an ordered tree representing its nested structure. An ordered tree \mathbf{T} is defined by a triplet (N, par, sib) , where $N = \{0, \dots, N\}$ is node indexes in the breadth first order (i.e., 0 is a root node), and $par : N \rightarrow N$ and $sib : N \rightarrow N$ define the parent and the previous sibling of each node, respectively. Thus, $par(n)$ denotes the n 's

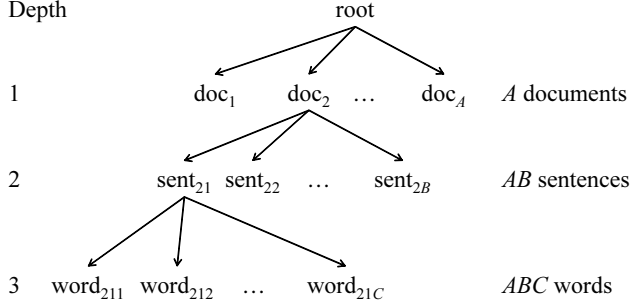


Figure 1: An ordered tree representation of a 3-layered nested structure.

parent and $sib(n)$ denotes the n 's previous sibling, where $sib(n)=0$ denotes that n has no previous sibling. Let D be the depth of \mathbf{T} and d_n be that of n . Also let N_d ($1 \leq d \leq D$) be a set of nodes n such that $d_n = d$. An observable variable x_n is associated with each node n , where variables in the same depth are assumed to share the same domain. For simplicity, we focus on the case that x_n ($n \in N_d$) has a discrete domain $\{1, \dots, V_d\}$, although our method can be extended easily to the continuous case, where $V_d = 0$ denotes that $n \in N_d$ has no observation.

We illustrate the ordered tree representation of a collection of documents with a 3-layered nested structure in Figure 1. Suppose that the collection contains A documents, where each document contains B sentences and each sentence consists of C words. A node in the first layer $n \in N_1$ corresponds to a document, and $n \in N_2$ and $n \in N_3$ correspond to a sentence and a word, respectively. The tree depth is $D=3$ and the tree size is $N = A + AB + ABC$. Each leaf node $n \in N_3$ has a corresponding observation x_n which is a term in V_3 -term vocabulary. Each inter node $n \in N \setminus N_3$ have no observation that is denoted by $V_1 = V_2 = 0$.

When we design a latent variable model for the above nested data, we choose essential structural information and reflect it in the model. In LDA models, for example, only document-level information is preserved and the other information (the sentence and word order) is ignored. In this paper, we aim to automatically extract important information in \mathbf{T} and generate an appropriate model for given nested data \mathbf{D} .

4 OUR MODEL

In this section, we first describe our latent variable model \mathbf{M} of which dependencies among latent variables are adjustable by its *structural parameters*. We then propose a local search method to optimize the parameters for given nested data \mathbf{D} . The method generates an appropriate model structure of \mathbf{M} that captures intrinsic characteristics of \mathbf{D} .

Table 1: An assumption variable A_d and dependencies.

A_d	Explanation	Dependency
I-det	Index-deterministic	$z_n = n$
P-det	Parent-deterministic	$z_n = z_\ell$
N-dep	Non-dependent	$z_n \perp\!\!\!\perp z_\ell, z_n \perp\!\!\!\perp z_m$
P-dep	Parent-dependent	$z_n \not\perp\!\!\!\perp z_\ell, z_n \perp\!\!\!\perp z_m$
S-dep	Sibling-dependent	$z_n \perp\!\!\!\perp z_\ell, z_n \not\perp\!\!\!\perp z_m$
B-dep	Both-dependent	$z_n \not\perp\!\!\!\perp z_\ell, z_n \not\perp\!\!\!\perp z_m$

4.1 MODEL DEFINITION

We here describe our model \mathbf{M} which is a Bayesian latent variable model for nested data $\mathbf{D} = (\mathbf{x}, \mathbf{T})$. Our model \mathbf{M} is defined by a quadruplet $(\mathbf{T}, \mathbf{A}, \boldsymbol{\alpha}, \boldsymbol{\beta})$, where \mathbf{T} is the given ordered tree, \mathbf{A} is *structural parameters* which control dependencies among latent variables, and $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are model parameters.

We first introduce latent variables $\mathbf{z} \equiv (z_n)_{n=1}^N$ and structural parameters $\mathbf{A} \equiv (A_d)_{d=1}^D$, where A_d is an *assumption variable* which defines dependencies among latent variables in the d th layer. Our model \mathbf{M} assumes that each node $n \in N_d$ has a discrete latent variable $z_n \in \{1, \dots, K_d\}$. We define $z_0 = 0$ for denoting the root 0 has no latent variable. Each observable variable x_n is generated by depending only on the corresponding latent variable z_n . Each latent variable z_n is generated by depending on latent variables of n 's parent and sibling $z_{par(n)}$ and $z_{sib(n)}$. For simplicity, let $\ell = par(n)$ and $m = sib(n)$ in this paper. An assumption variable A_d with a discrete domain $\{\text{I-det}, \text{P-det}, \text{N-dep}, \text{P-dep}, \text{S-dep}, \text{B-dep}\}$ controls the dependency of z_n ($n \in N_d$) as shown in Table 1. I-det and P-det denote that z_n deterministically takes a value n and z_ℓ , respectively. N-dep denotes that z_n is independent of other latent variables. P-dep, S-dep and B-dep denote that z_n depends on z_ℓ , z_m and the both, respectively. The structural parameters \mathbf{A} control the entire model structure of \mathbf{M} which represents dependencies among all latent variables \mathbf{z} .

We next introduce model parameters $\boldsymbol{\alpha} \equiv (\boldsymbol{\alpha}_d)_{d=1}^D$ and $\boldsymbol{\beta} \equiv (\boldsymbol{\beta}_d)_{d=1}^D$, where $\boldsymbol{\alpha}_d \equiv (\alpha_{d,k})_{k=1}^{K_d}$ and $\boldsymbol{\beta}_d \equiv (\beta_{d,v})_{v=1}^{V_d}$ are parameters of Dirichlet distributions for generating $\boldsymbol{\theta}_{d,i,j} \equiv (\theta_{d,i,j,k})_{k=1}^{K_d}$ and $\boldsymbol{\phi}_{d,k} \equiv (\phi_{d,k,v})_{v=1}^{V_d}$, respectively. Here, $\boldsymbol{\theta}_{d,i,j}$ is a parameter of a categorical distribution for generating z_n ($n \in N_d$) and $\boldsymbol{\phi}_{d,k}$ is that for generating x_n , that is, $\theta_{d,i,j,k}$ is a probability of $z_n = k$ given $z_\ell = i$ and $z_m = j$ and $\phi_{d,k,v}$ is a probability of $x_n = v$ given $z_n = k$. The generation process of $(\mathbf{x}, \mathbf{z}, \boldsymbol{\theta}, \boldsymbol{\phi})$ given model $\mathbf{M} \equiv (\mathbf{T}, \mathbf{A}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ is as follow.

1. For each depth $d = 1, \dots, D$ and hidden class $i = 1, \dots, K_{d-1}$ and $j, k = 1, \dots, K_d$
 - (a) Draw $\boldsymbol{\theta}_{d,i,j} \sim \text{Dir}(\boldsymbol{\alpha}_d)$
 - (b) Draw $\boldsymbol{\phi}_{d,k} \sim \text{Dir}(\boldsymbol{\beta}_d)$

2. For each depth $d=1, \dots, D$ and node $n \in N_d$

(a) Choose a class z_n by

case A_d
when I-det : $z_n := n$
when P-det : $z_n := z_\ell$
when N-dep : $z_n \sim \text{Cat}(\theta_{d,0,0})$
when P-dep : $z_n \sim \text{Cat}(\theta_{d,z_\ell,0})$
when S-dep : $z_n \sim \text{Cat}(\theta_{d,0,z_m})$
when B-dep : $z_n \sim \text{Cat}(\theta_{d,z_\ell,z_m})$

(b) Draw an observation $x_n \sim \text{Cat}(\phi_{d,z_n})$

Here, $\theta_{d,i,j}$ is a parameter for generating z_n ($n \in N_d$) given $z_\ell = i$ and $z_m = j$, however, we set $i = 0$ and $j = 0$ when z_n is independent of z_ℓ and z_m , respectively. The joint probability $p(\mathbf{x}, \mathbf{z}, \boldsymbol{\theta}, \boldsymbol{\phi} | \mathbf{M})$ factorizes into

$$p(\mathbf{x}, \mathbf{z}, \boldsymbol{\theta}, \boldsymbol{\phi} | \mathbf{M}) = p(\mathbf{x} | \mathbf{z}, \boldsymbol{\phi}) p(\mathbf{z} | \mathbf{A}, \boldsymbol{\theta}) p(\boldsymbol{\theta} | \boldsymbol{\alpha}) p(\boldsymbol{\phi} | \boldsymbol{\beta}),$$

and each probability is computed as

$$p(\boldsymbol{\theta} | \boldsymbol{\alpha}) = \prod_{d=1}^D \prod_{i=0}^{K_d-1} \prod_{j=0}^{K_d} \text{Dir}(\theta_{d,i,j}; \boldsymbol{\alpha}_d), \quad (1)$$

$$p(\boldsymbol{\phi} | \boldsymbol{\beta}) = \prod_{d=1}^D \prod_{k=1}^K \text{Dir}(\phi_{d,k}; \boldsymbol{\beta}_d), \quad (2)$$

$$p(\mathbf{z} | \mathbf{A}, \boldsymbol{\theta}) = \prod_{d=1}^D \prod_{i=0}^{K_d-1} \prod_{j=0}^{K_d} \prod_{k=1}^{K_d} \theta_{d,i,j,k}^{c_{d,i,j,k}(\mathbf{A}, \mathbf{z})}, \quad (3)$$

$$p(\mathbf{x} | \mathbf{z}, \boldsymbol{\phi}) = \prod_{d=1}^D \prod_{k=1}^{K_d} \prod_{v=1}^{V_d} \phi_{d,k,v}^{c_{d,k,v}(\mathbf{z}, \mathbf{x})}, \quad (4)$$

where $c_{d,i,j,k}(\mathbf{A}, \mathbf{z})$ and $c_{d,k,v}(\mathbf{z}, \mathbf{x})$ are define as

$$c_{d,i,j,k}(\mathbf{A}, \mathbf{z}) \equiv \begin{cases} |\{n \in N_d \mid z_n = k\}| & A_d = \text{N-dep}, i = j = 0 \\ |\{n \in N_d \mid z_\ell = i, z_n = k\}| & A_d = \text{P-dep}, j = 0 \\ |\{n \in N_d \mid z_m = j, z_n = k\}| & A_d = \text{S-dep}, i = 0 \\ |\{n \in N_d \mid z_\ell = i, z_m = j, z_n = k\}| & A_d = \text{B-dep} \\ 0 & \text{otherwise,} \end{cases}$$

$$c_{d,k,v}(\mathbf{z}, \mathbf{x}) \equiv |\{n \in N_d \mid z_n = k, x_n = v\}|.$$

Here, $c_{d,k,v}(\mathbf{z}, \mathbf{x})$ is the count of $\phi_{d,k,v}$ used in the generating process and $c_{d,i,j,k}(\mathbf{A}, \mathbf{z})$ is that of $\theta_{d,i,j,k}$.

Our model \mathbf{M} includes a variety of well-known existing models. Given text data \mathbf{x} and ordered tree \mathbf{T} shown in Figure 1 as its nested structure, we can represent various models by adjusting structural parameters \mathbf{A} as shown in Table 2, where the corresponding plate notations are shown in Figure 2.

Table 2: Example models for 3-layered text data. MMM denotes a multinomial mixture model. dX, sX and wX denote document-level X, sentence-level X and word-level X, respectively.

	$\mathbf{A} = (A_1, A_2, A_3)$	Corresponding Model
(1)	N-dep, P-det, P-det	dMMM
(2)	I-det, P-det, P-dep	dLDA
(3)	I-det, I-det, S-dep	wHMM
(4)	I-det, S-dep, P-dep	sHMM + wMMM
(5)	I-det, P-det, B-dep	dLDA + wHMM
(6)	I-det, B-dep, P-dep	dLDA + sHMM + wMMM

4.2 MODEL GENERATION

Given nested data $\mathbf{D} = (\mathbf{x}, \mathbf{T})$, we aim to generate the “best” model $\mathbf{M} = (\mathbf{T}, \mathbf{A}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ for \mathbf{D} by optimizing \mathbf{A} , $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. To define the “best”, a criterion for comparing multiple models is needed. A log marginal likelihood $\mathcal{L}[\mathbf{M}] \equiv \ln p(\mathbf{x} | \mathbf{M})$ is one of such criteria which measures how model \mathbf{M} fits to the given data \mathbf{x} [19, 10]. However, computing $\mathcal{L}[\mathbf{M}]$ is intractable because it requires an intractable summation $\sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z} | \mathbf{M})$. We instead employ the variational free energy (VFE) [5] $\mathcal{F}[\mathbf{M}]$, which is a lower bound of $\mathcal{L}[\mathbf{M}]$. The definition and an efficient computation algorithm of $\mathcal{F}[\mathbf{M}]$ are described in Section 5. The algorithm is also used for optimizing model parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ given \mathbf{D} and \mathbf{A} . Even though VFE $\mathcal{F}[\mathbf{M}]$ is computable, it is still intractable to maximize $\mathcal{F}[\mathbf{M}]$ with respect to \mathbf{A} because it requires evaluation of $\mathcal{F}[\mathbf{M}]$ for all possible instantiations for \mathbf{A} . We avoid such intractable evaluation by employing the following local search:

1. Let \mathbf{A} s.t. $A_d = \text{P-det}$ for all d and evaluate $\mathcal{F}[\mathbf{M}]$
2. Let $t := 0$ and $B_t := \{\mathbf{A}\}$
3. Let $NB_t :=$ all neighbors of $\mathbf{A} \in B_t$
4. Evaluate $\mathcal{F}[\mathbf{M}]$ for all $\mathbf{A} \in NB_t$
5. Let $B_{t+1} :=$ the w best structures w.r.t. VFE
6. Terminate if $B_{t+1} = B_t$ and return 3. otherwise

We initialize the first candidate \mathbf{A} as the simplest structure which contains only deterministic latent variables that is equivalent to no latent variable model. We then repeat generating neighbors of current candidates and choosing the w best structures as new candidates while VFE increases, where w is the search bandwidth. Here, \mathbf{A}' is a neighbor of \mathbf{A} if there exists exactly one d such that $A_d \neq A'_d$.

The above method automatically generates an appropriate latent variable model \mathbf{M} for the given nested data \mathbf{D} by optimizing its structural parameters \mathbf{A} and its model parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. It can also automatically select an appropriate K_d , which is the number of clusters in the d -th layer, by running with different K_d and choosing \mathbf{M} with the highest VFE. In the case we have a condition on models, our method can generate only expected models by skip-

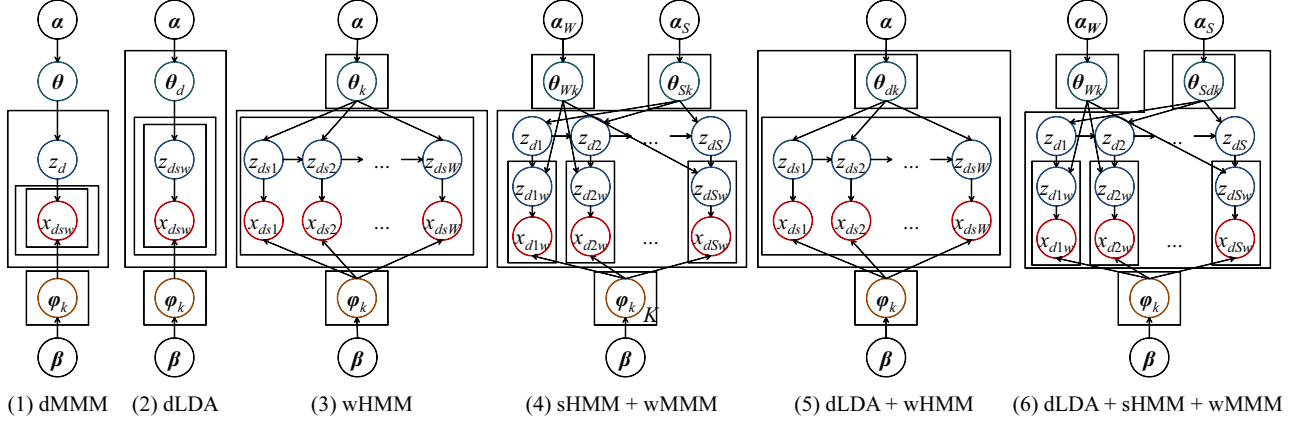


Figure 2: Plate representations of models in Table 2.

ping violated models in the search. For example, a model for the document clustering task should have a latent variable which indicates the cluster index of each document. Our method can generate a such model by excepting models which have no latent variable in the document-layer.

5 VARIATIONAL INFERENCE

We here define a variational free energy $\mathcal{F}[M]$ which is used as a measure how model M fits to given nested data D . We propose an efficient method for computing $\mathcal{F}[M]$ that employs a dynamic programming algorithm on T as its building block. The method is universal because it does not require change even if model structure A and data structure T are changed.

5.1 VARIATIONAL FREE ENERGY

Using Jensen's inequality, we obtain the following lower bound of log marginal likelihood $\mathcal{L}[M]$,

$$\begin{aligned} \ln p(\mathbf{x} | M) &\geq E_q[\ln p(\mathbf{x} | \mathbf{z}, \phi)] + E_q[\ln p(\mathbf{z} | \mathbf{A}, \theta)] \\ &\quad + E_q[\ln p(\theta | \alpha)] + E_q[\ln p(\phi | \beta)] - H[q] \\ &\equiv \mathcal{F}[q, M], \end{aligned}$$

where q is a variational distribution such that $q(\mathbf{z}, \theta, \phi) = q(\mathbf{z})q(\theta)q(\phi)$ and $H[q]$ is its entropy. By using the Euler-Lagrange equation, we obtain

$$q(\mathbf{z}) \propto \exp(E_{q(\phi)}[\ln p(\mathbf{x} | \mathbf{z}, \phi)] + E_{q(\theta)}[\ln p(\mathbf{z} | \theta)]), \quad (5)$$

$$q(\theta) \propto p(\theta | \alpha) \exp(E_{q(\mathbf{z})}[\ln p(\mathbf{z} | \mathbf{A}, \theta)]), \quad (6)$$

$$q(\phi) \propto p(\phi | \beta) \exp(E_{q(\mathbf{z})}[\ln p(\mathbf{x} | \mathbf{z}, \phi)]). \quad (7)$$

By iteratively updating q , we can maximize $\mathcal{F}[q, M]$ w.r.t. q . We can also maximize it w.r.t. model parameters α and β by fixed point iteration [21]. We use $\mathcal{F}[M]$ to denote $\mathcal{F}[q, M]$ which is computed by estimated q , α and β .

We next detail these q updates. Because $q(\theta)$ and $q(\phi)$ are approximations of (1) and (2), we define

$$q(\theta) \equiv \prod_{d=1}^D \prod_{i=0}^{K_d} \prod_{j=0}^{K_d} \text{Dir}(\theta_{d,i,j}; \mathbf{a}_{d,i,j}),$$

$$q(\phi) \equiv \prod_{d=1}^D \prod_{k=1}^K \text{Dir}(\phi_{d,k}; \mathbf{b}_{d,k}),$$

where $\mathbf{a}_{d,i,j} \equiv (a_{d,i,j,k})_{k=1}^{K_d}$ and $\mathbf{b}_{d,k} \equiv (b_{d,k,v})_{v=1}^{V_d}$ are variational parameters. By substituting (1)-(4) into (6)(7), their updates are obtained by

$$a_{d,i,j,k} = \alpha_{d,k} + E_{q(\mathbf{z})}[c_{d,i,j,k}(\mathbf{A}, \mathbf{z})], \quad (8)$$

$$b_{d,k,v} = \beta_{d,v} + E_{q(\mathbf{z})}[c_{d,k,v}(\mathbf{z}, \mathbf{x})]. \quad (9)$$

We also obtain the following updates of $q(\mathbf{z})$:

$$\begin{aligned} q(\mathbf{z}) &\propto \left(\prod_{d=1}^D \prod_{i=0}^{K_d-1} \prod_{j=0}^{K_d} \prod_{k=1}^{K_d} \theta_{d,i,j,k}^* c_{d,i,j,k}(\mathbf{A}, \mathbf{z}) \right) \\ &\quad \times \left(\prod_{d=1}^D \prod_{k=1}^{K_d} \prod_{v=1}^{V_d} \phi_{d,k,v}^* c_{d,k,v}(\mathbf{z}, \mathbf{x}) \right), \end{aligned}$$

where

$$\begin{aligned} \theta_{d,i,j,k}^* &\equiv \exp \left(\Psi(a_{d,i,j,k}) - \Psi \left(\sum_{k'=1}^{K_d} a_{d,i,j,k'} \right) \right), \\ \phi_{d,k,v}^* &\equiv \exp \left(\Psi(b_{d,k,v}) - \Psi \left(\sum_{v'=1}^{V_d} b_{d,k,v'} \right) \right), \end{aligned}$$

and $\Psi(\cdot)$ is a digamma function. Here, $q(\mathbf{z}) = p(\mathbf{z} | \mathbf{x}, \theta^*, \phi^*)^1$ holds by the definitions of (3) and (4).

¹Even $\phi_{d,k}^*$ and $\theta_{d,i,j}^*$ are not a probability vector, $p(\mathbf{z} | \mathbf{x}, \theta^*, \phi^*)$ is a probability through the effect of a normalizing factor $p(\mathbf{x} | \theta^*, \phi^*) = \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z} | \theta^*, \phi^*)$.

Thus, expectations in (8) and (9) are computed as

$$\mathbb{E}_{q(\mathbf{z})}[c_{d,i,j,k}(\mathbf{A}, \mathbf{z})] = \sum_{n \in N_d} P_{\ell,m,n}^{i,j,k}[\mathbf{x}, \boldsymbol{\theta}^*, \boldsymbol{\phi}^*] \quad (10)$$

$$\mathbb{E}_{q(\mathbf{z})}[c_{d,k,v}(\mathbf{z}, \mathbf{x})] = \sum_{n \in N_d: x_n=v} \sum_{i=0}^{K_d-1} \sum_{j=0}^{K_d} P_{\ell,m,n}^{i,j,k}[\mathbf{x}, \boldsymbol{\theta}^*, \boldsymbol{\phi}^*] \quad (11)$$

$$P_{\ell,m,n}^{i,j,k}[\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}] \equiv p(z_\ell=i, z_m=j, z_n=k \mid \mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}).$$

We propose an efficient dynamic programming method for computing $P_{\ell,m,n}^{i,j,k}[\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}]$ in Section 5.2.

5.2 EFFICIENT PROBABILITY COMPUTATION

For efficient computation of $P_{\ell,m,n}^{i,j,k}[\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}]$, we define

$$R_{\ell,m,n}^{i,j,k}[\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}] \equiv p(z_\ell=i, z_m=j, z_n=k, \mathbf{x} \mid \boldsymbol{\theta}, \boldsymbol{\phi}). \quad (12)$$

Using the above, $P_{\ell,m,n}^{i,j,k}[\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}]$ is computed as

$$P_{\ell,m,n}^{i,j,k}[\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}] = \frac{R_{\ell,m,n}^{i,j,k}[\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}]}{p(\mathbf{x} \mid \boldsymbol{\theta}, \boldsymbol{\phi})}, \quad (13)$$

$$p(\mathbf{x} \mid \boldsymbol{\theta}, \boldsymbol{\phi}) = \sum_{i=0}^{K_d-1} \sum_{j=0}^{K_d} R_{\ell,m,n}^{i,j,k}[\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}]. \quad (14)$$

Computing $R_{\ell,m,n}^{i,j,k}[\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}]$ by naively summarizing out $z_{n'}$ ($n' \in N \setminus \{\ell, m, n\}$) requires exponential time in N .

We here describe an efficient dynamic programming algorithm for computing $R_{\ell,m,n}^{i,j,k}[\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}]$. Let $\text{Dec}(n)$ be all descendants of n in \mathbf{T} . Also let $\text{Sib}^-(n)$ and $\text{Sib}^+(n)$ be all younger and older siblings of n , respectively. We then introduce the following four sets, *inside set* $\text{I}(n)$, *outside set* $\text{O}(n)$, *forward set* $\text{F}(n)$ and *backward set* $\text{B}(n)$:

$$\begin{aligned} \text{I}(n) &\equiv \{n\} \cup \text{Dec}(n), & \text{F}(n) &\equiv \bigcup_{m \in \{n\} \cup \text{Sib}^-(n)} \text{I}(m), \\ \text{O}(n) &\equiv N \setminus \text{Dec}(n), & \text{B}(n) &\equiv \bigcup_{m' \in \{n\} \cup \text{Sib}^+(n)} \text{I}(m'). \end{aligned}$$

Figure 3 shows examples of those sets of a 3-layered ordered tree. Using those sets, a set of all nodes N factorizes into $N = \text{O}(\ell) \cup \text{F}(m) \cup \text{B}(n)$. For a set $C \subseteq N$, we define $\mathbf{x}_C \equiv (x_n)_{n \in C}$ and $\mathbf{z}_C \equiv (z_n)_{n \in C}$. Then, $R_{\ell,m,n}^{i,j,k}[\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}]$ factorizes into

$$\begin{aligned} R_{\ell,m,n}^{i,j,k}[\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}] &= p(\mathbf{x}_{\text{O}(\ell)}, z_\ell=i \mid \boldsymbol{\theta}, \boldsymbol{\phi}) \\ &\quad p(\mathbf{x}_{\text{F}(m)}, z_m=j \mid z_\ell=i, \boldsymbol{\theta}, \boldsymbol{\phi}) \\ &\quad p(\mathbf{x}_{\text{B}(n)}, z_n=k \mid z_\ell=i, z_m=j, \boldsymbol{\theta}, \boldsymbol{\phi}). \end{aligned}$$

To compute the above, we introduce the following four probabilities, *inside probability* $I_n[k]$, *outside probability*

$O_n[k]$, *forward probability* $F_n[i, k]$ and *backward probability* $B_n[i, j, k]$:

$$I_n[k] \equiv p(x_{\text{I}(n)} \mid z_n=k, \boldsymbol{\theta}, \boldsymbol{\phi}),$$

$$O_n[k] \equiv p(x_{\text{O}(n)}, z_n=k \mid \boldsymbol{\theta}, \boldsymbol{\phi}),$$

$$F_n[i, k] \equiv p(x_{\text{F}(n)}, z_n=k \mid z_\ell=i, \boldsymbol{\theta}, \boldsymbol{\phi}),$$

$$B_n[i, j, k] \equiv p(x_{\text{B}(n)}, z_n=k, \mid z_\ell=i, z_m=j, \boldsymbol{\theta}, \boldsymbol{\phi}).$$

The above probabilities can be computed in the following dynamic programming manner:

$$I_n[k] = \phi_{d,k,v} B_c[k, 0], \quad (15)$$

$$O_n[k] = \sum_{i=0}^{K_d-1} \sum_{j=0}^{K_d} O_n[i, j, k], \quad (16)$$

$$F_n[i, k] = I_n[k] \sum_{j=1}^{K_d} F_m[i, j] \theta_{d,i,j,k}, \quad (17)$$

$$B_n[i, j, k] = I_n[k] B_{m'}[i, k] \theta_{d,i,j,k}, \quad (18)$$

where $v = x_n$, $d = d_n$, c is the oldest child of n , m' is the next sibling of n and

$$O_n[i, j, k] \equiv O_\ell[k] F_m[i, j] B_{m'}[i, j] \phi_{d,k,v} \theta_{d,i,j,k},$$

$$B_n[i, j] \equiv \sum_{k=1}^{K_d} B_n[i, j, k].$$

Finally, the target probability is computed by

$$R_{\ell,m,n}^{i,j,k}[\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}] = O_\ell[i] F_m[i, j] B_n[i, j, k]. \quad (19)$$

In summary, the variational free energy $\mathcal{F}[\mathbf{M}]$ is computed by Algorithm 1. When $K_d = K$ for all d , the complexity of this algorithm is $O(NK^3)$, however, it decreases according to structural parameters \mathbf{A} . For example, it becomes $O(NK^2)$ if \mathbf{A} represents an HMM, and also becomes $O(NK)$ if \mathbf{A} represents an LDA model. Note that the naive computation for $R_{\ell,m,n}^{i,j,k}[\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\phi}]$ requires exponential time in N but our method is polynomial.

6 EMPIRICAL RESULTS

6.1 ARTIFICIAL DATASETS

We here illustrate that our method is feasible for generating structures of latent variable models by using artificial datasets, where correct models which generate the datasets are known. We designed 12 models with the 3-layered nested structure \mathbf{T} shown in Figure 1 and generated 12 datasets from these models. Each dataset contains L documents, each document contains L sentences and each sentence contains L words. We set the cluster size of each depth as $K_1 = K_2 = K_3 = 5$ and the vocabulary size as $V_1 = V_2 = 0$ and $V_3 = 500$.

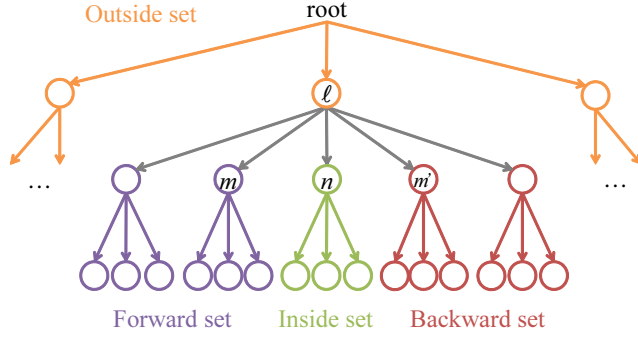


Figure 3: Inside set $I(n)$, outside set $O(\ell)$, forward set $F(m)$ and backward sets $B(m')$ of a 3-layered ordered tree, where m' is the next sibling of n .

Algorithm 1 Variational Free Energy $\mathcal{F}[M]$

```

repeat
  for  $n = 1, \dots, N$  do
    Compute  $I_n[k]$  for each  $k$  by (15)
    Compute  $B_n[i, j, k]$  for each  $i, j, k$  by (18)
  end for
  for  $n = N, \dots, 1$  do
    Compute  $F_n[i, k]$  for each  $i, k$  by (17)
    Compute  $O_n[k]$  for each  $k$  by (16)
  end for
  Compute  $R_{\ell, m, n}^{i, j, k}[\mathbf{x}, \boldsymbol{\theta}^*, \boldsymbol{\phi}^*]$  for each  $i, j, k$  by (19)
  Compute  $P_{\ell, m, n}^{i, j, k}[\mathbf{x}, \boldsymbol{\theta}^*, \boldsymbol{\phi}^*]$  for each  $i, j, k$  by (13)
  Compute  $E_{q(\mathbf{z})}[c_{d, i, j, k}(\mathbf{z})]$  for each  $d, i, j, k$  by (10)
  Compute  $E_{q(\mathbf{z})}[c_{d, k, v}(\mathbf{z})]$  for each  $d, k, v$  by (11)
  Update  $a_{d, i, j, k}$  for each  $d, i, j, k$  by (8)
  Update  $b_{d, k, v}$  for each  $d, k, v$  by (9)
  Update  $\alpha$  and  $\beta$  by fixed point iteration [21]
until  $\mathbf{a}, \mathbf{b}, \alpha$  and  $\beta$  converge
Return  $\mathcal{F}[q, M]$  computed by estimated  $\mathbf{a}, \mathbf{b}, \alpha, \beta$ 

```

We applied our method for the 12 datasets with search bandwidth $w = 3$. Table 3 shows the correct models and generated models. Our method generated the correct models for all dataset with $L = 50$. Even the dataset size is small, it correctly estimated simple models such as MMMs.

6.2 REAL-WORLD DATASETS

To demonstrate its applicability to real data, we applied our method for two real-world datasets, a Japanese blog dataset and an English news dataset. In both experiments, we ran our method with different $K = 10, 20, 30$ and chose the model with the highest VFE. We set search bandwidth $w = 3$ and excluded models of which complexity is $O(NK^3)$ for tractability. We also applied LDA models for the same datasets with different $K = 10, 20, \dots, 100$ and compared VFEs of LDA models and those of generated models.

Table 4: VFEs for the Japanese blog dataset.

Model	VFE
user-LDA	-5.239×10^6
article-LDA	-5.284×10^6
Generated model	-5.222×10^6

Table 6: VFEs for the English news dataset.

Model	First dataset	Second dataset
day-LDA	-8.739×10^6	-4.891×10^6
article-LDA	-8.299×10^6	-4.609×10^6
sentence-LDA	-8.554×10^6	-4.842×10^6
Generated model	-7.658×10^6	-4.555×10^6

JAPANESE BLOG DATASET. The dataset is a collection of 100 users’ goo blog [1] articles from April 7th to June 27th in 2007, where each article is extracted as a sequence of nouns and verbs. We filtered out the 100 most frequent terms as stop words and used the top 5,000 terms to construct a 3-layered nested dataset consisted of 100 users, 7,687 articles and 731,065 words.

For this dataset, our method generated the same model as (5) in Figure 2, which is equivalent to an HMTM [3] that is a combination of a user-level LDA and word-level HMM. The model is better than LDA models from the aspect of VFE as shown in Table 4. In the generated model, each user has a corresponding topic transition matrix but topics are shared by all users. Table 5 shows the ten most frequent topics and their five most probable words, where words were translated from Japanese to English. We manually gave a topic name for each topic. The table shows that the generated model captured latent topics in the Japanese blog dataset.

ENGLISH NEWS DATASET. The dataset is a collection of Reuters’ news articles in 1987 [2]. We extracted articles from March 1st to 31st and constructed a 4-layered nested dataset consisting of 29 days, 10,535 articles, 79,155 sentences and 31,057 terms in the vocabulary, where words including 0-9 were replaced to “NUM”. We then created two datasets by extracting the 5,000 most frequent terms: the first dataset that we did not filter out any words but the second one that we filtered the first 100 terms as stop words. Those datasets contained 1,369,888 and 626,316 words, respectively.

Figure 4 shows plate representations of models generated by our method. Our method generated a word-level HMM for the first dataset and a 3-layered MMM for the second one. Table 6 shows VFEs of the generated models and LDA models. As shown, our method found better models than LDA in terms of VFE.

Figure 5 shows the ten most frequent topics, their five most probable words and the five most probable topic transi-

Table 3: Correct models and generated models with $L = 10, 30, 50$. Incorrect assumptions are colored in red.

Correct Model		$L = 10$	$L = 30$	$L = 50$
1.	dMMM	N-dep, P-det, P-det	N-dep, P-det, P-det	N-dep, P-det, P-det
2.	sMMM	I-det, N-dep, P-det	I-det, N-dep, P-det	I-det, N-dep, P-det
3.	dLDA	I-det, P-det, P-dep	I-det, P-det, P-dep	I-det, P-det, P-dep
4.	sLDA	I-det, I-det, P-det	I-det, S-dep, P-det	I-det, I-det, P-dep
5.	dHMM	I-det , P-det, P-dep	S-dep, P-det, P-det	S-dep, P-det, P-det
6.	sHMM	I-det, S-dep, P-det	I-det, S-dep, P-det	I-det, S-dep, P-det
7.	wHMM	P-det , P-det, P-det	N-dep, P-det, B-dep	N-dep, P-det, B-dep
8.	dHMM + wMMM	I-det , P-det, P-dep	I-det , P-det, P-dep	S-dep, P-det, P-dep
9.	sHMM + wMMM	P-det , P-det , P-det	N-dep , B-dep , P-dep	I-det, S-dep, P-dep
10.	dLDA + sHMM	P-det , S-dep, P-det	S-dep , B-dep , P-det	I-det, P-det, B-dep
11.	dLDA + wHMM	P-det , P-det , P-det	S-dep , P-det, B-dep	I-det, B-dep, P-det
12.	sLDA + wHMM + wMMM	P-det , P-det , P-det	S-dep , B-dep, P-det	I-det, B-dep, P-dep

Table 5: The ten most frequent topics and their five most probable words obtained from the Japanese blog data.

z_{uaw}	Topic name	The five most probable words
24	Verbs 1	sleep, go, try, mind, no
5	Stock market	stock, management, company, proportion, market
16	News	book, ads, news, US, company
9	Research	psychology, perception, research, knowledge, description
28	Animation	version, animation, appearance, track, purchase
27	Travel (urban)	bus, station, Rahmen, father, taste
12	Travel (nature)	blossom, weather, Japanese cherry, park, wind
25	Software	case, screen, data, process, layout
19	International news	dictionary, multimedia, phraseology, terrorism, high concept
8	Verbs 2	receive, book, visit, try, time

tions over those topics. The generated model captured frequently-appearing grammatical patterns. A topic transition ($18 \rightarrow 7$) describes a pattern (number \rightarrow unit), and ($\{29, 13\} \rightarrow 26 \rightarrow \{17, 22\}$) describes a pattern (Preposition (1) & (2) \rightarrow Article \rightarrow {Objects, Adjective}). Because such grammatical patterns commonly appear in every sentences, our method extracted the word-layer and removed the other layers.

In contrast, our method generated a 3-layered MMM (day-article-sentence) for the second dataset. This is because grammatical patterns in the dataset were destroyed by filtering out stop words. The model performed day, article and sentence clustering; days were grouped into two clusters and articles and sentences were grouped into 30 clusters. Table 7 shows the three most probable article and sentence clusters and the five most probable words. The day clusters shared neither article cluster nor sentence cluster in the table. Sentence clusters in day cluster 3 describe periodic financial reports, budgets of IT companies and shareholder meeting, respectively. Those in day cluster 26 describe policies of executives, domestic economy and economic relations between Japan and US, respectively. It seems that day cluster 3 tends to mention about periodical events.

7 CONCLUSION

We proposed a method for generating latent variable models from nested data. We presented a latent variable model of which structure is adjustable by its structural parameters. Our model attempts to optimize the parameters by maximizing the variational free energy. We derived a universal variational inference method for our model based on dynamic programming on the given data structure. No matter how deep the input nested data are, our method extracts its essential information and generates a model with an appropriate complexity. We empirically showed that our method correctly generated model structures by synthetic datasets and also showed that it extracted hidden structures of blog and news datasets.

We note a potential direction for future work. We can extend our model to non-parametric Bayesian setting by replacing Dirichlet distributions by HDPs [23]. To the best of our knowledge, it seems difficult to provide an efficient variational inference method for the extended model because it subsumes the infinite hidden Markov model (iHMM) [4] which is a non-parametric Bayesian extension of HMM. However, we can easily derive a Gibbs sampling in the same way as that of iHMM. The beam sampling [25] is an efficient sampling method for iHMM, which com-

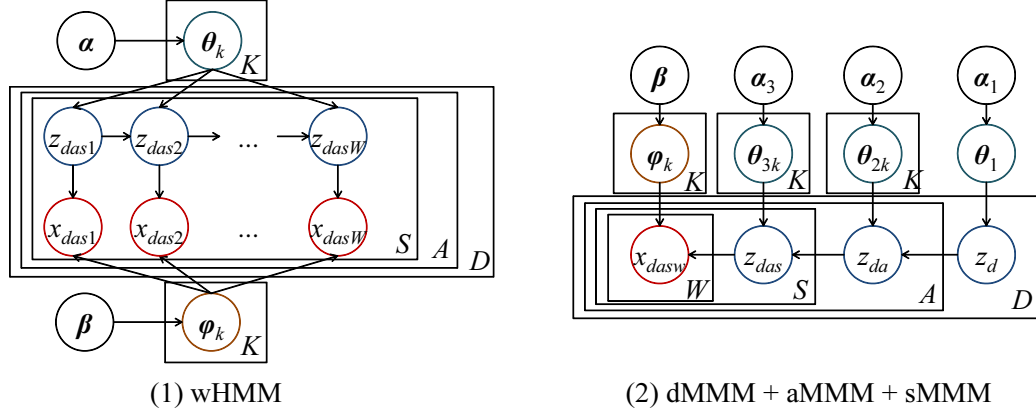


Figure 4: Plate representations of generated models from two English News datasets.

z_{dasw}	Topic Name	The five most probable words
26	Articles	the, a, its, an, this
29	Prepositions (1)	in, to, for, on, by
22	Objects	company, year, bank, week, government
18	Numbers	NUM, two, one, five, three
17	Adjective	new, us, first, common, united
14	Verb (1)	will, is, would, has, was
13	Prepositions (2)	of, in, for, and, from
4	Economic words	oil, foreign, trade, exchange, economic
7	Units	mln, pct, billion, cts, dlrs
19	Verb (2)	said, that, told, added, but

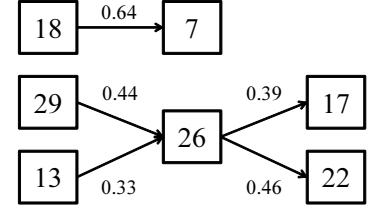


Figure 5: Obtained topics and topic translations by word-level HMM from the English news dataset.

Table 7: Obtained topics by 3-layered MMM from the English news dataset.

day cluster z_d	The three most probable clusters		The five most probable words
	article cluster z_{da}	sentence cluster z_{das}	
3	25	6	prior, record, div, pay, qtlly
		19	revs, oper, note, avg, shrs
		29	quarter, earnings, reported, ended, tax
	26	27	management, plant, unit, selling, underwriting
		22	computer, system, products, software, data
		20	owned, subsidiary, unit, plc, agreed
	23	5	common, shareholders, dividend, board, record
		1	offer, proceeds, stake, common, capital
		27	management, plant, unit, selling, underwriting
26	28	16	buffer, strike, minister, party, union
		2	president, chief, executive, chairman, officer
		11	there, think, don't, no, do
	19	26	world, loans, payments, economic, countries
		11	there, think, don't, no, do
		3	tax, growth, rate, budget, deficit
	27	9	industry, companies, financial, markets, business
		28	japan, japanese, minister, president, reagan
		26	world, loans, payments, economic, countries

bins slice sampling and dynamic programming. It would be interesting to extend it for the extended model by intro-

ducing our dynamic programming algorithm.

References

- [1] Goo blog. <http://blog.goo.ne.jp>.
- [2] Reuters-21578 text categorization test collection. <http://www.daviddlewis.com/resources/testcollections/reuters21578/>.
- [3] Mark Andrews and Gabriella Vigliocco. The Hidden Markov Topic Model: A Probabilistic Model of Semantic Representation. *Topics in Cognitive Science*, 2(1):101–113, January 2010.
- [4] MJ Beal. The infinite hidden Markov model. In *Proc. NIPS*, 2001.
- [5] M.J. Beal and Z. Ghahramani. The Variational Bayesian EM Algorithm for Incomplete Data: with Application to Scoring Graphical Model Structures. *Bayesian Statistics*, 7, 2003.
- [6] David M Blei, Lawrence Carin, and David Dunson. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, November 2012.
- [7] David M Blei, AY Ng, and MI Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.
- [8] DM Chickering. Learning Bayesian networks is NP-complete. In *Learning from Data: Artificial Intelligence and Statistics*. 1996.
- [9] DM Chickering, D Heckerman, and C Meek. Large-sample learning of Bayesian networks is NP-hard. *JMLR*, 5:1287–1330, 2004.
- [10] Gregory F. Cooper and Edward Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–347, October 1992.
- [11] Lan Du, Wray Buntine, and Huidong Jin. A segmented topic model based on the two-parameter Poisson-Dirichlet process. *Machine learning*, 81(1):5–19, July 2010.
- [12] Roger Grosse and RR Salakhutdinov. Exploiting compositionality to explore a large space of model structures. In *Proc. UAI*, 2012.
- [13] Amit Gruber, Y Weiss, and M Rosen-Zvi. Hidden topic Markov models. In *Proc. AISTATS*, 2007.
- [14] Thomas Hofmann. Probabilistic latent semantic analysis. In *Proc. UAI*, 1999.
- [15] Tomoharu Iwata, Tsutomu Hirao, and Naonori Ueda. Unsupervised Cluster Matching via Probabilistic Latent Variable Models. In *Proc. AAAI*, 2013.
- [16] Tomoharu Iwata, T Yamada, and N Ueda. Probabilistic latent semantic visualization: topic model for visualizing documents. In *Proc. KDD*, 2008.
- [17] Changhe Yuan James Cussens, Brandon Malone. Tutorial on Optimal Algorithms for Learning Bayesian Networks. In *IJCAI 2013 Tutorial*, 2013.
- [18] Do-kyum Kim, G Voelker, and LK Saul. A Variational Approximation for Topic Modeling of Hierarchical Corpora. In *Proc. ICML*, 2013.
- [19] David J C Mackay. Bayesian interpolation. *Neural computation*, 4(3):415–447, May 1992.
- [20] Brandon Malone and C Yuan. Evaluating anytime algorithms for learning optimal Bayesian networks. In *Proc. UAI*, 2013.
- [21] Thomas P Minka. Estimating a Dirichlet distribution, 2000. <http://research.microsoft.com/minka/papers/dirichlet>.
- [22] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, California: Morgan Kaufmann, 1988. 2nd printing edition.
- [23] YW Teh and MI Jordan. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.
- [24] Marc Teyssier and D Koller. Ordering-based search: A simple and effective algorithm for learning Bayesian networks. In *Proc. UAI*, 2005.
- [25] Jurgen Van Gael, Yunus Saatci, Yee Whye Teh, and Zoubin Ghahramani. Beam sampling for the infinite hidden Markov model. In *Proc. ICML*, 2008.
- [26] Su-in Lee Varun and Ganapathi Daphne. Efficient Structure Learning of Markov Networks using L1-Regularization. In *Proc. NIPS*, 2006.
- [27] Changhe Yuan, Brandon Malone, and Xiaojian Wu. Learning optimal Bayesian networks using A* search. In *Proc. IJCAI*, 2011.
- [28] NL Zhang. Hierarchical latent class models for cluster analysis. *JMLR*, 5:697–723, 2004.

Monotone Closure of Relaxed Constraints in Submodular Optimization: Connections Between Minimization and Maximization

Rishabh Iyer

Dept. of Electrical Engineering
University of Washington
Seattle, WA-98175, USA

Stefanie Jegelka

Dept. of EECS
University of California, Berkeley
Berkeley, CA-94720, USA

Jeff Bilmes

Dept. of Electrical Engineering
University of Washington
Seattle, WA-98175, USA

Abstract

It is becoming increasingly evident that many machine learning problems may be reduced to submodular optimization. Previous work addresses generic discrete approaches and specific relaxations. In this work, we take a generic view from a relaxation perspective. We show a relaxation formulation and simple rounding strategy that, based on the monotone closure of relaxed constraints, reveals analogies between minimization and maximization problems, and includes known results as special cases and extends to a wider range of settings. Our resulting approximation factors match the corresponding integrality gaps. For submodular maximization, a number of relaxation approaches have been proposed. A critical challenge for the practical applicability of these techniques, however, is the complexity of evaluating the multilinear extension. We show that this extension can be efficiently evaluated for a number of useful submodular functions, thus making these otherwise impractical algorithms viable for real-world machine learning problems.

1 INTRODUCTION

Submodularity is a natural model for many real-world problems including many in the field of machine learning. Submodular functions naturally model aspects like cooperation, complexity, and attractive potentials in minimization problems, and also notions of diversity, coverage, and information in maximization problems. A function $f : 2^V \rightarrow \mathbb{R}$ on subsets of a ground set $V = \{1, 2, \dots, n\}$ is *submodular* [37, 15] if for all subsets $S, T \subseteq V$, we have $f(S) + f(T) \geq f(S \cup T) + f(S \cap T)$. The *gain* of an element $j \in V$ with respect to $S \subseteq V$ is defined as $f(j|S) \triangleq f(S \cup j) - f(S)$. Submodularity is equivalent to *diminishing gains*: $f(j|S) \geq f(j|T), \forall S \subseteq T, j \notin T$.

A large number of machine learning problems may be

phrased as submodular minimization or maximization problems. In this paper, we address the following two very general forms of submodular optimization:

$$\text{Problem 1: } \min_{X \in \mathcal{C}} f(X), \quad \text{Problem 2: } \max_{X \in \mathcal{C}} f(X)$$

Here, \mathcal{C} denotes a family of feasible sets, described e.g., by cardinality constraints, or by combinatorial constraints insisting that the solution be a tree, path, cut, matching, or a cover in a graph.

Applications. Unconstrained submodular minimization occurs in machine learning and computer vision in the form of combinatorial regularization terms for sparse reconstruction and denoising, and MAP inference, e.g. for image segmentation [30]. Other applications are well modeled as constrained submodular minimization. For example, a rich class of models for image segmentation has been encoded as minimizing a submodular functions subject to cut constraints [28]. Similarly, [9] efficiently solves MAP inference in a sparse higher-order graphical model through submodular vertex cover, and [48] proposes to interactively segment images by minimizing a submodular function subject to connectivity constraints, i.e., the selected set of vertices must contain an s - t path. Moreover, bounded-complexity corpus construction [36] can be modeled as cardinality constrained submodular minimization. Constrained submodular maximization is a fitting model for problems such as optimal sensing [32], marketing [29], document summarization [35], and speech data subset selection [34].

Previous Work. Since most instances of Problems 1 and 2 are NP-hard, one must strive for approximations that have bounded error. Broadly speaking¹, the algorithms can be classified into discrete (*combinatorial*) and continuous *relaxation* based. The discrete approaches were initially proposed for certain *specific* constraints [17, 27, 47, 41, 12, 4, 3], but later made *general* and unified [25, 18, 24]. In the case of submodular minimization, the discrete approaches have been based on approximating the submodular function

¹ Emphasized words in this paragraph correspond to headings in Table 1, which also serves as a summary.

by tractable approximations [25, 18], while in the case of submodular maximization, they have been based on greedy and local search techniques [25, 41, 12, 4, 3]. Most of these algorithms are *fast* and scalable. The continuous relaxation techniques, on the other hand, have so far either been analyzed for very specific constraints, or when general, are too *slow* to use in practice. For example, in the case of minimization, they were studied only for the specific constraints of covers [20] and cuts [27], and in the case of maximization, the techniques though general have yet to show significant practical impact due to their prohibitive computational costs [6, 5]. Hence discrete algorithms are typically used in applications (e.g., [34]).

Constraints or Function	Operation (& speed)	Algorithm Approach	
		Combinatorial	Relaxation
Specific	Min (fast)	[17, 27]	[20, 27]
	Min (slow)	[47]	Unnecessary
	Max (fast)	[41, 12, 4, 3]	This paper
	Max (slow)	Unnecessary	[4, 5]
General	Min (fast)	[25]	This paper
	Min (slow)	[18]	Unnecessary
	Max (fast)	[25]	Open
	Max (slow)	Unnecessary	[6]

Table 1: Past work & our contributions (see text for explanation).

In the present paper, we develop a continuous relaxation methodology for Problems 1 and 2 that applies not only for multiple types of constraints but that even establishes connections between minimization and maximization problems. We summarize our contributions, in comparison to previous work, in Table 1, which lists one problem as being still open, and other problems as being unnecessary (given a “fast” approach, the corresponding “slow” approach is unnecessary). Our techniques are not only connective, but also fast and scalable. In the case of constrained minimization, we provide a formulation applicable for a large class of constraints. In the case of submodular maximization, we show how for a large class of submodular functions of practical interest, the generic slow algorithms can be made fast and scalable. We note, however, that it is still an open problem to provide a fast and scalable algorithmic framework (with theoretical guarantees) based on continuous relaxations for general submodular maximization.

The connections between minimization and maximization is based on the up- or down-monotonicity of the constraint set: up-monotone constraints are relevant for submodular minimization problems, and down-monotone constraints are relevant for submodular maximization problems. Our relaxation viewpoint, moreover, complements and improves on the bounds found in [25]. For example, where [25] may have an approximation bound of k , our results imply a bound of $n - k + 1$, where $n = |V|$, so considering both [25] and our new work presented here, we obtain combined bounds of the form $\min(k, n - k + 1)$ (more specifics are given in Table 2). This also holds for maximization – in certain cases discrete

algorithms obtain suboptimal results, while relaxation techniques obtain improved, and sometimes optimal guarantees.

The idea of our relaxation strategy is as follows: the submodular function $f(S)$, which is defined on the vertices of the n -dimensional hypercube, is extended to a function defined on $[0, 1]^n$. The two functions valuate identically if the vector $x \in [0, 1]^n$ is the characteristic vector of a set. We then solve a continuous optimization problem subject to linear constraints. For minimization, the convex *Lovász extension* defined in Eqn. (1) is a suitable extension of f . Appropriately rounding the resulting optimal continuous solutions leads to a number of approximation guarantees. For maximization, ideally we could utilize a concave extension. Since the tightest concave extension of a submodular function is hard to characterize [49], we instead use the *multilinear extension* (see Eqn. (2)) that behaves like a concave function in certain directions [6, 5]. Our resulting algorithms often achieve better bounds than discrete greedy approaches.

Paper Roadmap. For constrained minimization (Sec. 3), we provide a generic approximation factor (Theorem 1), for the general class of constraints defined in Eq. 4. We show that many important constraints, including matroid, cardinality, covers, cuts, paths, matchings, etc. can be expressed as Eq. 4. As a corollary to our main result (Theorem 1), we obtain known results (like covers [20] and cuts [27]), and also novel ones (for spanning trees, cardinality constraints, paths, matchings etc.). We also show bounds on integrality gaps for constrained submodular minimization, which to our knowledge is novel. In the context of maximization (Sec. 4), we provide closed form multi-linear extensions for several submodular functions useful in applications. We also discuss the implications of these algorithmically. Note that this is particularly important, given that many optimal algorithms for several submodular maximization problems are based on the multilinear extension. Lastly, we extend our techniques to minimize the difference between submodular functions, and provide efficient optimization and rounding techniques for these problems (Sec. 5).

2 CONTINUOUS RELAXATIONS

Convex relaxation. The Lovász extension [37] reveals an important connection between submodularity and convexity, and is defined as follows. For each $y \in [0, 1]^n$, we obtain a permutation σ_y by ordering its elements in non-increasing order, and thereby a chain of sets $\Sigma_0^y \subseteq \dots \subseteq \Sigma_n^y$, with $\Sigma_j^y = \{\sigma_y(1), \dots, \sigma_y(j)\}$ for $j \in \{1, 2, \dots, n\}$. The Lovász extension \check{f} of f is a weighted sum of the ordered entries of y :

$$\check{f}(y) = \sum_{j=1}^n y[\sigma_y(j)] (f(\Sigma_j^y) - f(\Sigma_{j-1}^y)) \quad (1)$$

The Lovász extension is unique (despite possibly non-unique orderings if y has duplicate entries), and convex if and only if f is submodular. Since it agrees with f on the vertices of the hypercube, i.e., $f(X) = \tilde{f}(1_X)$, for all $X \subseteq V$ (where 1_X is the characteristic vector of X , i.e., $1_X(j) = I(j \in X)$), \tilde{f} is a natural convex extension of a submodular function. The Lovász extension is a non-smooth (piece-wise linear) convex function for which a subgradient $h_{\sigma_y}^f$ at y can be computed efficiently via Edmonds's greedy algorithm [10]:

$$h_{\sigma_y}^f(\sigma_y(j)) = f(\Sigma_j^y) - f(\Sigma_{j-1}^y), \quad \forall j \in \{1, 2, \dots, n\}$$

The Lovász extension has also found applications in defining norms for structured sparsity [1] and divergences for rank aggregation [23].

Multilinear relaxations. For maximization problems, the relaxation of choice has frequently been the multilinear extension [12]

$$\tilde{f}(x) = \sum_{X \subseteq V} f(X) \prod_{i \in X} x_i \prod_{i \notin X} (1 - x_i), \quad (2)$$

where f is any set function. Since Eqn. (2) has an exponential number of terms, its evaluation in general computation-ally expensive, or requires approximation.

One may define at least two types of gradients for the multilinear extension. The first, “standard” gradient is

$$\nabla_j \tilde{f}(x) = \partial \tilde{f} / \partial x_j = \tilde{f}(x \vee e_j) - \tilde{f}(x \vee e_j - e_j).$$

where $e_j = 1_{\{j\}}$, and $\{x \vee y\}(i) = \max(x(i), y(i))$. A second gradient is $\nabla_j^a \tilde{f}(x) = \tilde{f}(x \vee e_j) - \tilde{f}(x)$. The two gradients are related component-wise as $\nabla_j \tilde{f}(x) = (1 - x_j) \nabla_j^a \tilde{f}(x)$, and both can be computed in $O(n)$ evaluations of \tilde{f} .

Optimization. Relaxation approaches for submodular optimization follow a two-stage procedure:

1. Find the optimal (or approximate) solution \hat{x} to the problem $\min_{x \in \mathcal{P}_C} \tilde{f}(x)$ (or $\max_{x \in \mathcal{P}_C} \tilde{f}(x)$).
2. Round the continuous solution \hat{x} to obtain the discrete indicator vector of set \hat{X} .

Here, \mathcal{P}_C denotes the polytope corresponding to the family \mathcal{C} of feasible sets – i.e., their convex hull or its approximation, which is a “continuous relaxation” of the constraints \mathcal{C} . The final approximation factor is then $f(\hat{X})/f(X^*)$, where X^* is the exact optimizer of f over \mathcal{C} .

An important quantity is the *integrality gap* that measures – over the class \mathcal{S} of all submodular (or monotone submodular) functions – the largest possible discrepancy between the optimal discrete solution and the optimal continuous solution. For minimization problems, the integrality gap is defined as:

$$\mathcal{I}_C^{\mathcal{S}} \triangleq \sup_{f \in \mathcal{S}} \frac{\min_{X \in \mathcal{C}} f(X)}{\min_{x \in \mathcal{P}_C} \tilde{f}(x)} \geq 1. \quad (3)$$

For maximization problems, we would take the supremum over the inverse ratio. In both cases, $\mathcal{I}_C^{\mathcal{S}}$ is defined only for non-negative functions. The integrality gap largely depends on the specific formulation of the relaxation. Intuitively, it provides a lower bound on our approximation factor: we usually cannot expect to improve the solution by rounding, because any rounded discrete solution is also a feasible solution to the relaxed problem. One rather only hopes, when rounding, to not worsen the cost relative to that of the continuous optimum. Indeed, integrality gaps can often be used to show tightness of approximation factors obtained from relaxations and rounding [7]. For a detailed discussion on this connection, see [26].

3 SUBMODULAR MINIMIZATION

For submodular minimization, the optimization problem in Step 1 is a convex optimization problem, and can be solved efficiently if one can efficiently project onto the polytope \mathcal{P}_C . Our second ingredient is rounding. To round, a surprisingly simple thresholding turns out to be quite effective for a large number of constrained and unconstrained submodular minimization problems: choose an appropriate $\theta \in (0, 1)$ and pick all elements with “weights” above θ , i.e., $\hat{X}_\theta = \{i : \hat{x}(i) \geq \theta\}$. We call this procedure the *θ -rounding procedure*. In the following sections, we first review relaxation techniques for unconstrained minimization (which are known), and afterwards phrase a generic framework for constrained minimization. Interestingly, both constrained and unconstrained versions essentially admit the same rounding strategy and algorithms.

3.1 UNCONSTRAINED MINIMIZATION

Continuous relaxation techniques for unconstrained submodular minimization have been well studied [1, 15]. In this case, $\mathcal{P}_C = [0, 1]^n$, and importantly, the approximation factor and integrality gap are both 1.

Lemma 1. [15] *For any submodular function f , it holds that $\min_{X \subseteq V} f(X) = \min_{x \in [0, 1]^n} \tilde{f}(x)$. Given a continuous minimizer $x^* \in \operatorname{argmin}_{x \in [0, 1]^n} \tilde{f}(x)$, the discrete minimizers are exactly those obtained by θ -rounding x^* , for any $\theta \in (0, 1)$.*

Since the Lovász extension is a non-smooth convex function, it can be minimized up to an additive accuracy of ϵ in $O(1/\epsilon^2)$ iterations of the subgradient method. This accuracy directly transfers to the discrete solution if we choose the best set obtained with any $\theta \in (0, 1)$ [1]. For special cases, such as submodular functions derived from concave functions, smoothing techniques yield a convergence rate of $O(1/t)$ [45].

3.2 CONSTRAINED MINIMIZATION

We next address submodular minimization under constraints, where rounding affects the accuracy of the discrete solution. By appropriately formulating the problem, we show that θ -rounding applies to a large class of problems. We assume that the family \mathcal{C} of feasible solutions can be expressed by a (low-order) polynomial number of linear inequalities, or at least that linear optimization over \mathcal{C} can be done efficiently, as is the case for matroid polytopes [10].

A straightforward relaxation of \mathcal{C} is the convex hull $\mathcal{P}_{\mathcal{C}} = \text{conv}(1_X, X \in \mathcal{C})$ of \mathcal{C} . Often however, it is not possible to obtain a decent description of the inequalities determining $\mathcal{P}_{\mathcal{C}}$, even in cases when minimizing a linear function over \mathcal{C} is easy (two examples are the s-t cut and s-t path polytopes [44]). In those cases, we relax \mathcal{C} to its *up-monotone* closure $\hat{\mathcal{C}} = \{X \cup Y \mid X \in \mathcal{C} \text{ and } Y \subseteq V\}$. With $\hat{\mathcal{C}}$, a set is feasible if it is in \mathcal{C} or a superset of a set in \mathcal{C} . The convex hull of $\hat{\mathcal{C}}$ is the up-monotone extension of $\mathcal{P}_{\mathcal{C}}$ within the hypercube, i.e. $\mathcal{P}_{\hat{\mathcal{C}}} = \hat{\mathcal{P}}_{\mathcal{C}} = (\mathcal{P}_{\mathcal{C}} + \mathbb{R}_+^n) \cap [0, 1]^n$, which is often easier to characterize than $\mathcal{P}_{\mathcal{C}}$ [26]. If \mathcal{C} is already up-monotone, then $\hat{\mathcal{P}}_{\mathcal{C}} = \mathcal{P}_{\mathcal{C}}$.

Optimization. The relaxed minimization problem $\min_{x \in \hat{\mathcal{P}}_{\mathcal{C}}} \check{f}(x)$ is non-smooth and convex with linear constraints, and therefore amenable to, e.g., projected subgradient methods. We here assume that the submodular function f is monotone nondecreasing (which often holds in applications), and extend our results to non-monotone functions in [26].

For projected (sub)gradient methods, it is vital that the projection on $\hat{\mathcal{P}}_{\mathcal{C}}$ can be done efficiently. Indeed, this holds with the above assumptions that we can efficiently solve a linear optimization over $\hat{\mathcal{P}}_{\mathcal{C}}$. In this case, e.g. Frank-Wolfe [13] methods apply. The projection onto matroid polyhedra can also be cast as a form of unconstrained submodular function minimization and is hence polynomial time solvable [15]. To apply splitting methods such as the alternating directions method of multipliers (ADMM) [2], we write the problem as $\min_{x, y: x=y} \check{f}(x) + I(y \in \hat{\mathcal{P}}_{\mathcal{C}})$. ADMM needs a projection oracle onto the constraints – discussed above – and the proximal operator of f . Computing the proximal operator of the Lovász extension is equivalent to unconstrained submodular minimization, or to solving the minimum norm point problem. In special cases, faster algorithms apply [39, 45].

Rounding. Once we have obtained a minimizer \hat{x} of \check{f} over $\hat{\mathcal{P}}_{\mathcal{C}}$, we apply simple θ -rounding. Whereas in the unconstrained case, \hat{X}_{θ} is feasible for any $\theta \in (0, 1)$, we must now ensure $\hat{X}_{\theta} \in \hat{\mathcal{C}}$. Hence, we pick the largest threshold θ such that $\hat{X}_{\theta} \in \hat{\mathcal{C}}$, i.e., the smallest \hat{X}_{θ} that is feasible. This is always possible since $\hat{\mathcal{C}}$ is up-monotone and contains V . The threshold θ can be found using $O(\log n)$ checks among the sorted entries of the continuous solution \hat{x} . The following lemma states how the threshold θ determines

a worst-case approximation:

Lemma 2. *For a monotone submodular f and any $\hat{x} \in [0, 1]^V$ and $\theta \in (0, 1)$ such that $\hat{X}_{\theta} = \{i \mid \hat{x}_i \geq \theta\} \in \hat{\mathcal{C}}$, it holds that $f(\hat{X}_{\theta}) \leq \frac{1}{\theta} \check{f}(\hat{x})$. If, moreover, $\check{f}(\hat{x}) \leq \beta \min_{x \in \hat{\mathcal{P}}_{\mathcal{C}}} \check{f}(x)$, then it holds that $f(\hat{X}_{\theta}) \leq \frac{\beta}{\theta} \min_{X \in \mathcal{C}} f(X)$.*

The set \hat{X}_{θ} is in $\hat{\mathcal{C}}$ and therefore guaranteed to be a superset of a solution $\hat{Y}_{\theta} \in \mathcal{C}$. As a final step, we prune down \hat{X}_{θ} to $\hat{Y}_{\theta} \subseteq \hat{X}_{\theta}$. Since the objective function is nondecreasing, $f(\hat{Y}_{\theta}) \leq f(\hat{X}_{\theta})$, Lemma 2 holds for \hat{Y}_{θ} as well. If, in the worst case, $\theta = 0$, then the approximation bound in Lemma 2 is unbounded. Fortunately, in most cases of interest we obtain polynomially bounded approximation factors.

In the following, we will see that our $\hat{\mathcal{P}}_{\mathcal{C}}$ provides the basis for relaxation schemes under a variety of constraints, and that these, together with θ -rounding, yield bounded-factor approximations. We assume that there exists a family $\mathcal{W} = \{W_1, W_2, \dots\}$ of sets $W_i \subseteq V$ such that the polytope $\hat{\mathcal{P}}_{\mathcal{C}}$ can be described as

$$\hat{\mathcal{P}}_{\mathcal{C}} = \left\{ x \in [0, 1]^n \mid \sum_{i \in W} x_i \geq b_W \text{ for all } W \in \mathcal{W} \right\}. \quad (4)$$

Analogously, this means that $\hat{\mathcal{C}} = \{X \mid |X \cap W| \geq b_W, \text{ for all } W \in \mathcal{W}\}$. In our analysis, we do not require \mathcal{W} to be of polynomial size, but a linear optimization over $\hat{\mathcal{P}}_{\mathcal{C}}$ or a projection onto it should be possible at least within a bounded approximation factor. This is the case for s-t paths and cuts, covering problems, and spanning trees.

The following main result (proven in [26]) states approximation bounds and integrality gaps for the class of problems described by Equation (4).

Theorem 1. *The θ -rounding scheme for constraints \mathcal{C} whose relaxed polytope $\hat{\mathcal{P}}_{\mathcal{C}}$ can be described by Equation (4) achieves a worst case approximation bound of $\max_{W \in \mathcal{W}} |W| - b_W + 1$.*

We also show [26] that this factor matches the integrality gap for the constraints considered in this paper.

A result similar to Theorem 1 was shown in [31] for a different, greedy algorithmic technique. While their result also holds for a large class of constraints, for the constraints in Equation (4) they obtain a factor of $\max_{W \in \mathcal{W}} |W|$, which is worse than Theorem 1 if $b_W > 1$. This is the case, for instance, for matroid span constraints, cardinality constraints, trees and multiset covers.

Pruning. The final piece of the puzzle is the pruning step, where we reduce the set $\hat{X}_{\theta} \in \hat{\mathcal{C}}$ to a final solution $\hat{Y}_{\theta} \subseteq \hat{X}_{\theta}$ that is feasible: $\hat{Y}_{\theta} \in \mathcal{C}$. This is important when the true constraints \mathcal{C} are not up-monotone, as is the case for cuts or paths. Since we have assumed that the function f is

monotone, pruning can only reduce the objective value. The pruning step means finding *any* subset of \hat{X}_θ that is in \mathcal{C} , which is often not hard. We propose the following heuristic for this: if \mathcal{C} admits (approximate) linear optimization, as is the case for all the constraints considered here, then we may improve over a given rounded subset by assigning additive weights: $w(i) = \infty$ if $i \notin \hat{X}_\theta$, and otherwise use either uniform ($w(i) = 1$) or non-uniform ($w(i) = 1 - \hat{x}(i)$) weights. We then solve $\hat{Y}_\theta \in \operatorname{argmin}_{Y \in \mathcal{C}} \sum_{i \in Y} w(i)$. Uniform weights lead to the solution with minimum cardinality, and non-uniform weights will give a bias towards elements with higher certainty in the continuous solution. Truncation via optimization works well for paths, cuts, matchings or matroid constraints. In the extended version [26], we discuss how to handle non-monotone submodular functions and down-monotone constraints.

To demonstrate the utility of Theorem 1, we apply it to a variety of problems. We state only the results, all proofs are in [26]. Many of the constraints below are based on a graph $G = (\mathcal{V}, \mathcal{E})$, and in that case the ground set is the set \mathcal{E} of graph edges. When the context is clear, we overload notation and refer to $n = |\mathcal{V}|$ and $m = |\mathcal{E}|$. Results are summarized in Table 2.

3.2.1 MATROID CONSTRAINTS

An important class of constraints are matroid span or base constraints, with cardinality constraints (uniform matroids) and spanning trees (graphic or cycle matroids) as special cases. A matroid $\mathcal{M} = (\mathcal{I}_\mathcal{M}, r_\mathcal{M})$ is defined by its down-monotone family of independent sets $\mathcal{I}_\mathcal{M}$ or its rank function $r_\mathcal{M} : 2^V \rightarrow \mathbb{R}$. A set Y is a *spanning set* if its rank is that of V : $r_\mathcal{M}(Y) = r_\mathcal{M}(V)$. It is a *base* if $|Y| = r_\mathcal{M}(Y) = r_\mathcal{M}(V)$. Hence, the family of all spanning sets is the up-monotone closure of the family of all bases (e.g., supersets of spanning trees of a graph in the case of a graphic matroid). See [44] for more details on matroids. Let $\mathcal{S}_\mathcal{M}$ denote the spanning sets of matroid \mathcal{M} , and set $k = r_\mathcal{M}(V)$. It is then easy to see that with $\mathcal{C} = \mathcal{S}_\mathcal{M}$, the polytope $\mathcal{P}_\mathcal{C}$ is the matroid span polytope, which can be described as $\mathcal{P}_\mathcal{C} = \{x \in [0, 1]^n, x(S) \geq r_\mathcal{M}(V) - r_\mathcal{M}(V \setminus S), \forall S \subseteq V\}$ [44]. This is clearly in the form of Eqn. 4. Although this polytope is described via an exponential number of inequalities, it can be projected onto efficiently via submodular minimization [15].

Corollary 1. *Let \hat{Y}_θ be the rounded and pruned solution obtained from minimizing the Lovász extension over the span polytope. Then $f(\hat{Y}_\theta) \leq (n - k + 1)f(X^*)$. The integrality gap is also $n - k + 1$.*

In general, the rounding step will only provide an \hat{X}_θ that is a spanning set, but not a base. We can prune it to a base by greedily finding a maximum weight base among the elements of \hat{X}_θ . The worst-case approximation factor

of $n - k + 1$ complements other known results for this problem [25, 18]. The semi-gradient framework of [25] guarantees a bound of k , while more complex (and less practical) approximations [18] yield factors of $O(\sqrt{n})$. The factor k of [25] is the best for small k , while our continuous relaxation works well when k is large.

Cardinality Constraints. This is a special class of a matroid, called the uniform matroid. Since it suffices to analyze monotone submodular functions, the constraint of interest is $\mathcal{C} = \{X : |X| = k\}$. In this case, the corresponding polytope takes a very simple form: $\mathcal{P}_\mathcal{C} = \{x \in [0, 1]^n : \sum_i x_i = k\}$. Furthermore, the rounding step in this context is very intuitive. It corresponds to choosing the elements with the k largest entries in \hat{x} .

Spanning Trees. Here, the ground set $V = \mathcal{E}$ is the edge set in a graph and \mathcal{C} is the set of all spanning trees. The corresponding polytope $\mathcal{P}_\mathcal{C}$ is then the spanning tree polytope. Our bound in this setting is $m - n + 1$. The discrete algorithms of [25, 17] achieve a complementary bound of $|\mathcal{V}| = n$. For dense graphs, the discrete algorithms admit better worst case guarantees, while for sparse graphs (e.g., embeddable into r -regular graphs for small r), our guarantees are better.

3.2.2 SET COVERS

A fundamental family of constraints are set covers. Given a universe \mathcal{U} , and a family of sets $\{S_i\}_{i \in V}$, the task is to find a subset $X \subseteq V$ that covers the universe, i.e., $\bigcup_{i \in X} S_i = \mathcal{U}$, and has minimum cost as measured by a submodular function $f : 2^S \rightarrow \mathbb{R}$. The set cover polytope is up-monotone, constitutes the set of fractional covers, and is easily represented by Eqn. (4) as $\mathcal{P}_\mathcal{C} = \{x \in [0, 1]^{|\mathcal{V}|} \mid \sum_{i: u \in S_i} x(i) \geq 1, \forall u \in \mathcal{U}\}$. The following holds for minimum submodular set cover:

Corollary 2. *The approximation factor of our algorithm, and the integrality gap for the minimum submodular set cover problem, is $\gamma = \max_{u \in \mathcal{U}} |\{i : u \in S_i\}|$.*

The approximation factor in Corollary 2 (without the integrality gap) was first shown in [20]. The quantity γ corresponds to the maximum frequency of the elements in \mathcal{U} .

A generalization of set cover is the multi-set cover problem [43], where every element u is to be covered multiple (c_u) times. The multi-cover constraints can be formalized as $\mathcal{P}_\mathcal{C} = \{x \in [0, 1]^{|\mathcal{S}|} \mid \sum_{i: u \in S_i} x(i) \geq c_u, \forall u \in \mathcal{U}\}$.

Corollary 3. *The approximation factor and integrality gap of the multi-set cover problem is $\max_{u \in \mathcal{U}} |\{i : u \in S_i\}| - c_u + 1$.*

This result also implies the bound for set cover (with $c_u = 1$). Since the rounding procedure above yields a solution that is already a set cover (or a multi set cover), a subsequent pruning step is not necessary.

²These results were shown in [17, 20, 47]

	Matroid Constraints		Set Covers		Paths, Cuts and Matchings		
	Cardinality	Trees	Vertex Covers	Edge Covers	Cuts	Paths	Matchings
CR.	$n - k + 1$	$m - n + 1$	2	$\deg(G) \leq n$	$P_{max} \leq n$	$C_{max} \leq m$	$\deg(G) \leq n$
SG	k	n	$ VC \leq n$	$ EC \leq n$	$C_{max} \leq m$	$P_{max} \leq n$	$ M \leq n$
EA	\sqrt{n}	\sqrt{m}	\sqrt{n}	\sqrt{m}	\sqrt{m}	\sqrt{m}	\sqrt{m}
Integrality Gaps	$\Omega(n - k + 1)$	$\Omega(m - n + 1)$	2	$\Omega(n)$	$\Omega(n)$	$\Omega(m)$	$\Omega(n)$
Hardness ²	$\Omega(\sqrt{n})$	$\Omega(n)$	$2 - \epsilon$	$\Omega(n)$	$\Omega(\sqrt{m})$	$\Omega(n^{2/3})$	$\Omega(n)$

Table 2: Comparison of the results of our framework (CR) with the semigradient framework of [25] (SG), the Ellipsoidal Approximation (EA) algorithm of [18], hardness [17, 20, 47], and the integrality gaps of the corresponding constrained submodular minimization problems. Note the complementarity between CR and SG.

Vertex Cover. A vertex cover is a special case of a set cover, where \mathcal{U} is the set of edges in a graph, V is the set of vertices, and S_v is the set of all edges incident to $v \in V$. Corollary 2 implies a 2-approximation for submodular vertex cover, which matches the integrality gap and the lower bound in [17]. The 2-approximation for vertex cover was also shown in [17, 20].

Edge Cover. In the Edge Cover problem, \mathcal{U} is the set of vertices in a graph, V is the set of edges and S_v contains the two vertices comprising edge v . We aim to find a subset of edges such that every vertex is covered by some edge in the subset. It is not hard to see that the approximation factor we obtain is the maximum degree of the graph $\deg(G)$, which is upper bounded by $|\mathcal{V}|$, but is often much smaller. The algorithm in [25] has an approximation factor of the size of the edge cover $|EC|$, which is also upper bounded by $O(|\mathcal{V}|)$. These factors match the lower bound shown in [17].

3.2.3 CUTS, PATHS AND MATCHINGS

Even though Eqn. (4) is in the form of covering constraints, it can help solve problems with apparently very different types of constraints. The covering generalization works if we relax \mathcal{C} to its up-monotone closure: $\hat{\mathcal{C}}$ demands that a feasible set must contain (or “cover”) a set in \mathcal{C} . To go from $\hat{\mathcal{C}}$ back to \mathcal{C} , we prune in the end.

Cuts and Paths. Here, we aim to find an edge set $X \subseteq \mathcal{E}$ that forms an s-t path (or an s-t cut), and that minimizes the submodular function f . Both the s-t path and s-t cut polytopes are hard to characterize. However, their up-monotone extension $\hat{\mathcal{P}}_{\mathcal{C}}$ can be easily described. Furthermore, both these polytopes are intimately related to each other as a blocking pair of polyhedra (see [44]). The extended polytope for s-t paths can be described as a *cut-cover* [44] (i.e., any path must hit every cut at least once): $\hat{\mathcal{P}}_{\mathcal{C}} = \{x \in [0, 1]^{|\mathcal{E}|} \mid \sum_{e \in C} x(e) \geq 1, \text{ for every s-t cut } C \subseteq \mathcal{E}\}$. The closure of the s-t path constraint (or the cut-cover) is also called s-t connectors [44]. Conversely, the extended s-t cut polytope can be described as a *path-cover* [44, 27]: $\hat{\mathcal{P}}_{\mathcal{C}} = \{x \in [0, 1]^{|\mathcal{E}|} \mid \sum_{e \in P} x(e) \geq 1, \text{ for every s-t path } P \subseteq \mathcal{E}\}$.

Corollary 4. *The relaxation algorithm yields an approximation factor of $P_{max} \leq |\mathcal{V}|$ and $C_{max} \leq |\mathcal{E}|$ for minimum submodular s-t path and s-t cut respectively (P_{max} and*

C_{max} refer to the maximum size simple s-t path and s-t cut respectively). These match the integrality gaps for both problems.

While the description of the constraints as covers reveals approximation bounds, it does not lead to tractable algorithms for minimizing the Lovász extension. However, the extended cut and the extended path polytopes can be described exactly by a linear number of inequalities [42, 44]. The pruning step for paths and covers becomes a shortest path or minimum cut problem, respectively. As in the other cases, the approximations obtained from relaxations complement the bounds of P_{max} for paths and C_{max} for cuts shown in [25].

Perfect Matchings. Given a graph $G = (\mathcal{V}, \mathcal{E})$, the goal is to find a set of edges $X \subseteq \mathcal{E}$, such that X is a perfect matching in G and minimizes the submodular function f . For a bipartite graph, the polytope $\hat{\mathcal{P}}_{\mathcal{C}}$ can be characterized as $\mathcal{P}_{\mathcal{C}} = \{x \in [0, 1]^{|\mathcal{E}|} \mid \sum_{e \in \delta(v)} x(e) = 1 \text{ for all } v \in \mathcal{V}\}$, where $\delta(v)$ denotes the set of edges incident to v . Similar to the case of Edge Cover, Theorem 1 implies an approximation factor of $\deg(G) \leq |\mathcal{V}|$, which matches the lower bound shown in [17, 24].

4 SUBMODULAR MAXIMIZATION

To relax submodular maximization, we use the multilinear extension. We first show that this extension can be efficiently computed for a large subclass of submodular functions (deferring detailed derivations to [26]). As above, \mathcal{C} denotes the family of feasible sets, and $\mathcal{P}_{\mathcal{C}}$ the polytope corresponding to \mathcal{C} . For maximization, it makes sense to consider \mathcal{C} to be down-monotone (particularly when the function is monotone). Such a down-monotone \mathcal{C} could represent for example, matroid independence constraints, or upper bounds on the cardinality $\mathcal{C} = \{X : |X| \leq k\}$. Analogous to the case of minimization [26], an approximation algorithm for down-monotone constraints can be extended to up-monotone constraints, by using $f'(X) = f(V \setminus X)$.

The relaxation algorithms use the multilinear extension (Eqn. (2)) which in general requires repeated sampling and can be very expensive to compute. In the below, we show how this can be computed efficiently and exactly for many

practical and useful submodular functions.

Weighted Matroid Rank functions. A common class of submodular functions are sums of weighted matroid rank functions, defined as: $f(X) = \sum_i \max\{w_i(A) | A \subseteq X, A \in \mathcal{I}_i\}$, for linear weights $w_i(j)$. These functions form a rich class of coverage functions for summarization tasks [34]. The multilinear extension can be efficiently computed for a number of specific instances of this function. One such special case is the facility location objective [34]: $f(X) = \sum_{i \in V} \max_{j \in X} s_{ij}$, for pairwise similarities s_{ij} . The facility location function admits a nice representation of the multilinear extension (the full derivation is in [26]): $\tilde{f}(x) = \sum_{i \in V} \sum_{l=1}^n s_{ij_i^l} x_{ij_i^l} \prod_{m=1}^l (1 - x_{ij_i^m})$, where for each $i \in V$, the indices $j_i^1, j_i^2, \dots, j_i^n$ denote in sorted order the elements closest to i (in terms of similarity s_{ij}). We can similarly obtain closed form expressions with other forms of matroids, including uniform matroids or partition matroids. Due to limited space, detailed derivations are all given in [26].

Set Cover function: This is another important function, capturing notions of coverage [34]. Given a set of sets $\{\mathcal{S}_1, \dots, \mathcal{S}_n\}$ and the universe $\mathcal{U} = \cup_i \mathcal{S}_i$, define $f(X) = w(\cup_{i \in X} \mathcal{S}_i)$, where w_j denotes the weight of item $j \in \mathcal{U}$. This setup can alternatively be expressed via a neighborhood function $\Gamma : 2^V \rightarrow 2^{\mathcal{U}}$ such that $\Gamma(X) = \cup_{i \in X} \mathcal{S}_i$. Then $f(X) = w(\Gamma(X))$. Let $\Gamma^{-1}(j) = \{i \in V : j \in \Gamma(i)\}$. Then the multilinear extension has a simple form: $\tilde{f}(x) = \sum_{j \in \mathcal{U}} w_j [1 - \prod_{i \in \Gamma^{-1}(j)} (1 - x_i)]$. Again, for full derivations see [26].

Probabilistic Coverage Functions. This is a generalization of the set cover function above, and has been used in a number of models for summarization problems [11]. This provides a probabilistic notion to the set cover function, and can be defined as $f(X) = \sum_{i \in \mathcal{U}} w_i [1 - \prod_{j \in X} (1 - p_{ij})]$. We get back the set cover function, when p_{ij} is a binary vector (either i covers j or not). This function also has an efficiently computable multilinear extension [26]: $\tilde{f}(x) = \sum_{i \in \mathcal{U}} w_i [1 - \prod_{j \in V} (1 - p_{ij} x_j)]$.

Graph Cut related functions Graph cuts are a widely used class of functions. Their multilinear extension also admits closed form representation. The function and its multilinear extension can be written as: $f(X) = \sum_{i \in X, j \notin X} s_{ij}$, $\tilde{f}(x) = \sum_{i,j \in V} s_{ij} x_i (1 - x_j)$. A related function is a similarity penalizing function: $f(X) = -\sum_{i,j \in X} s_{ij}$. This function has been used for encouraging diversity [35, 34]. Its multilinear extension is $\tilde{f}(x) = -\sum_{i,j \in V} s_{ij} x_i x_j$. The detailed derivations of both these expressions are in [26].

Sparse Pseudo-Boolean functions. For graphical models, in particular in computer vision, set functions are often

written as polynomials [19]. Any set function can be written as a polynomial, $p_f(x) = \sum_{T \subseteq V} \alpha_T \prod_{i \in T} x_i$, where $x \in \{0, 1\}^n$ is the characteristic vector of a set. In other words, $f(S) = \sum_{T \subseteq S} \alpha_T$. Submodular functions are a subclass of these polynomials. This representation directly gives the multilinear extension as the same polynomial, $\tilde{f}(x) = \sum_{T \subseteq V} \alpha_T \prod_{i \in T} x_i$, and is efficiently computable if the polynomial is *sparse*, i.e., has few nonzero coefficients α_T [26]. This is the case for graph cut like functions above and for the functions considered in [46, 19]. We have been unable to find the above result elsewhere in the literature, so we formalize it as follows:

Proposition 1. *The polynomial representation is the multilinear extension: $\tilde{f}(x) = p_f(x)$*

Spectral functions. Diversity can also be encouraged via spectral regularizers [8]. Given a positive definite matrix $S \in \mathbb{R}^{n \times n}$, define S_X to be the $|X| \times |X|$ sub-matrix of the rows and columns indexed by X . Any scalar function ψ whose derivative is operator-antitone defines a submodular function, $f(X) = \sum_{i=1}^{|X|} \psi(\lambda_i(S_X))$, by applying it to the eigenvalues of S_X [14]. The resulting class of submodular functions includes the log determinants occurring in DPP inference [16], and, more generally, a smoothed log-determinant function $f(X) = \log \det(S_X + \delta I_X) = \sum_{i=1}^{|X|} \log(\lambda_i(S_X) + \delta)$. It is monotone for $\delta \geq 1$, and has an efficiently computable soft-max extension that is similar to the multilinear extension [16]. A related function that encourages diversity is $f(X) = -\sum_{i=1}^{|X|} (\lambda_i(S_X) - 1)^2$ [8]. It has a surprisingly simple multilinear extension: $\tilde{f}(x) = -\sum_{i,j \in V} s_{ij}^2 x_i x_j + \sum_{i \in V} (2s_{ii} + 1)$. For the detailed derivation of this, see [26].

Given expressions for the functions above, we can also handle weighted combinations $f(X) = \sum_i \lambda_i f_i(X)$, since its multilinear extension is $\tilde{f}(x) = \sum_i \lambda_i \tilde{f}_i(x)$. In the following sections, we briefly describe relaxation algorithms and rounding schemes for maximization.

4.1 MONOTONE MAXIMIZATION

We first investigate monotone submodular maximization subject to matroid independence constraints \mathcal{I} . The technique for maximizing the multilinear extension is the continuous greedy algorithm [50], which is a slight modification of the Frank-Wolfe algorithm [13], with a fixed step size. In each iteration, the algorithm takes a step $x^{t+1} = x^t + \delta h^t$ (with step size $\delta = 1/n^2$) in the direction $h^t = \arg\max_{h' \in \mathcal{P}_C} \langle h', \nabla^a \tilde{f}(x^t) \rangle$ best aligned with the alternate gradient. This *continuous greedy* procedure terminates in $O(n^2)$ iterations, after which we are guaranteed to obtain a point x such that $\tilde{f}(x) \geq (1 - 1/e) \tilde{f}(x^*)$ [5, 49]. Moreover, using the pipage rounding technique (in particular, the deterministic variant [50]) ensures that we can round the continuous solution to a set in $O(n^2)$ function calls.

³This extends to top- k facility location as well.

⁴This is for soft-max extension [16].

	Fac. Location ³	Set Cover	Graph Cuts	Diversity I/ II	Concave over card.	log-Det ⁴
Multilinear Closed form	$O(n^3)$	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n^3)$
Multilinear Sampling	$O(n^7 \log n)$	$O(n^6)$	$O(n^7)$	$O(n^7)$	$O(n^6)$	$O(n^8)$
Gradient Closed form	$O(n^3)$	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n^3)$
Gradient Sampling	$O(n^7 \log n)$	$O(n^7)$	$O(n^8)$	$O(n^8)$	$O(n^7)$	$O(n^9)$

Table 3: Complexity of evaluating the multilinear extensions and their gradients for both the optimized closed forms given in this paper and for sampling at high accuracy.

Unfortunately, naïve computation of the multilinear extension or its gradient takes exponential time. To compute these in polynomial time, we can apply sampling techniques. To obtain an accuracy better than $1/n^2$, we need $O(n^5)$ samples for the multilinear extension or for each coordinate of its gradient [50, 49]. This implies a complexity of $O(n^6)$ function evaluations for the gradient and $O(n^5)$ function evaluations for the extension itself, thus implying the algorithm’s complexity as $O(n^8 T_{\nabla f})$, where $T_{\nabla f}$ is the time of evaluating the gain of f . For facility location, this means a running time of $O(n^9 \log n)$, and for set cover functions $O(n^9)$.

But these high complexities are for using a sampling approximation of the generic expression for the multilinear extension (Eqn. (2)). The specialized expressions in Section 4 lead to algorithms that run several orders of magnitude faster. With $O(n^2)$ iterations, the time becomes $O(n^2 T_{\nabla \tilde{f}})$, where $\nabla \tilde{f}$ is the time to compute the gradient of \tilde{f} . Table 3 compares the function evaluation times for some useful submodular functions. Moreover, we can also use mixtures of these submodular functions, each with efficiently computable multilinear extensions, and compute the resulting multilinear extension also efficiently. While this is still slower than the accelerated greedy [38], it gains power for more complex constraints, such as matroid independence constraints, where the discrete greedy algorithm only achieves an approximation factor of $1/2$, whereas the continuous greedy obtains at least a $1 - 1/e$ factor. Similarly, the continuous greedy algorithm achieves a $1 - 1/e$ approximation guarantee for multiple knapsack constraints [33], while the discrete greedy techniques do not have such guarantees. Hence, the formulations above make it possible to use the optimal theoretical results with a more manageable running time.

4.2 NON-MONOTONE MAXIMIZATION

In the non-monotone setting, we must find a local optimum of the multilinear extension. We could use, for example, a Frank-Wolfe style algorithm [13] and run it until it converges to a local optimum. It is easy to see that at convergence x satisfies $\langle \nabla \tilde{f}(x), y - x \rangle \leq 0, \forall y \in \mathcal{P}_C$ and is a local optimum. Practically, this would mean checking if $\arg\max_{y \in \mathcal{P}_C} \langle y, \nabla \tilde{f}(x) \rangle = x$. For simple or no constraints, we could also use a method like L-BFGS. Running this procedure twice, we are guaranteed to obtain a 0.25 approximate solution [6]. This procedure works for any

down-monotone constraint \mathcal{C} . Moreover, this procedure with a slightly different extension has been successfully applied in practice to MAP inference with determinantal point processes [16].

A generic rounding strategy for submodular maximization problems was given by [6], and works for a large class of constraints (including matroid, knapsack constraints, and a combination thereof). Without constraints, this amounts to sampling a set by a distribution based on the continuous solution x — it will satisfy $\mathbf{E}_{X \sim x} f(X) = \tilde{f}(x)$. In practice, however, this may not work well. Since the multilinear extension is linear in any coordinate (holding the other ones fixed), a simpler co-ordinate ascent scheme of choosing the better amongst 0 or 1 for any fractional co-ordinate will guarantee a deterministic procedure of obtaining an integral solution no worse than the continuous one.

The above algorithms and rounding techniques offer a general and optimal framework, even for many complex constraints. Moreover, many of the best algorithms for non-monotone submodular maximization are based on the multilinear extension. For example, the best known algorithm for cardinality constrained non-monotone submodular maximization [4] uses a continuous double greedy algorithm on the multilinear extension. However, the practical utility of those algorithms is heavily impaired by computational complexity. In fact, non-monotone functions even require $O(n^7)$ samples [6]. For DPPs, [16] used an extension that is practical and close to the multilinear extension. Since they do not use the multilinear extension, the above rounding schemes do not imply the same approximation bounds as for the multilinear extension, leaving the worst-case approximation quality unknown. The expressions we show above use the multilinear extension and maintain its benefits, demonstrating that for many functions of practical interest, sampling, and hence extremely high complexity, is not necessary. This observation is a step from theory into practice, and allows for the improved approximations to occur in practice.

4.3 INTEGRALITY GAPS

Surprisingly, the multilinear extension has an integrality gap of 1 for a number of constraints including the matroid and cardinality constraints, since it is easy to round it exactly (using say, the pipage rounding or contention resolution schemes [5, 6]).

5 DIFFERENCE OF SUBMODULAR (DS) FUNCTIONS

Finally, we investigate minimizing the differences between submodular functions. Given submodular functions f and g , we consider the following minimization problem: $\min_{X \in \mathcal{C}} (f(X) - g(X))$. In fact, any set function can be represented as a difference between two non-negative monotone submodular functions [40, 21]. In the unconstrained setting, $\mathcal{C} = 2^V$. A natural continuous relaxation (not necessarily convex) is $\tilde{h}(x) = \check{f}(x) - \check{g}(x)$. The continuous relaxation problem is a DC programming problem, and can be addressed (often very efficiently) using the convex-concave procedure [51]. Moreover, thanks to the special structure of the Lovász extension, there exists a simple rounding scheme for the unconstrained version.

Lemma 3. *Given submodular functions f and g , and a continuous vector x , there exists a $\theta \in (0, 1)$ such that $f(X_\theta) - g(X_\theta) \geq \check{f}(x) - \check{g}(x)$, where $X_\theta = \{x \geq \theta\}$. Moreover, the integrality gap of $\tilde{h}(x)$ (in the unconstrained setting) is equal to 1.*

6 DISCUSSION

We have provided a unifying view to continuous relaxation methods for submodular optimization. For minimization problems with various constraints, we provide a generic rounding strategy with new approximation bounds and matching integrality gaps. For maximization, we provide efficiently computable expressions for many practically interesting submodular functions. This is a useful step towards transferring optimal theoretical results to real-world applications. An interesting question remains whether there exist improved sampling schemes for cases where the multilinear extension is too complex. Also recently, [22] investigated forms of submodular minimization and maximization, with submodular constraints. The proposed algorithms there were all discrete, and it will be interesting if our framework could extend to their setting as well.

Acknowledgments: We thank Bethany Herwaldt, Karthik Narayanan, Kai Wei and the rest of the submodular group at UW for discussions. This material is based upon work supported by the National Science Foundation under Grant No. (IIS-1162606), and is also supported by a Google, a Microsoft, and an Intel research award. SJ's work is supported by the Office of Naval Research under contract/grant number N00014-11-1-0688, and gifts from Amazon Web Services, Google, SAP, Blue Goji, Cisco, Clearstory Data, Cloudera, Ericsson, Facebook, General Electric, Hortonworks, Intel, Microsoft, NetApp, Oracle, Samsung, Splunk, VMware and Yahoo!.

References

[1] F. Bach. *Learning with Submodular functions: A convex Optimization Perspective*, volume 6 of *Foundations and Trends in Machine Learning*. 2013.

[2] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[3] N. Buchbinder, M. Feldman, J. Naor, and R. Schwartz. A tight $(1/2)$ linear-time approximation to unconstrained submodular maximization. In *FOCS*, 2012.

[4] N. Buchbinder, M. Feldman, J. Naor, and R. Schwartz. Submodular maximization with cardinality constraints. In *SODA*, 2014.

[5] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.

[6] C. Chekuri, J. Vondrák, and R. Zenklus. Submodular function maximization via the multilinear relaxation and contention resolution schemes. *STOC*, 2011.

[7] E. Chlamtac and M. Tulsiani. Convex relaxations and integrality gaps. In *Handbook on Semidefinite, Conic and Polynomial Optimization*, pages 139–169. Springer, 2012.

[8] A. Das, A. Dasgupta, and R. Kumar. Selecting diverse features via spectral regularization. In *NIPS*, 2012.

[9] A. Delong, O. Veksler, A. Osokin, and Y. Boykov. Minimizing sparse high-order energies by submodular vertex-cover. In *NIPS*, 2012.

[10] J. Edmonds. Submodular functions, matroids and certain polyhedra. *Combinatorial structures and their Applications*, 1970.

[11] K. El-Arini, G. Veda, D. Shahaf, and C. Guestrin. Turning down the noise in the blogosphere. In *KDD*, 2009.

[12] U. Feige, V. Mirrokni, and J. Vondrák. Maximizing non-monotone submodular functions. *SIAM J. COMPUT.*, 40(4):1133–1155, 2007.

[13] M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval research logistics quarterly*, 1956.

[14] S. Friedland and S. Gaubert. Submodular spectral functions of principal submatrices of a hermitian matrix, extensions and applications. *Linear Algebra and its Applications*, 2011.

[15] S. Fujishige. *Submodular functions and optimization*, volume 58. Elsevier Science, 2005.

[16] J. Gillenwater, A. Kulesza, and B. Taskar. Near-optimal MAP inference for determinantal point processes. In *NIPS*, 2012.

[17] G. Goel, C. Karande, P. Tripathi, and L. Wang. Approximability of combinatorial problems with multi-agent submodular cost functions. In *FOCS*, 2009.

[18] M. Goemans, N. Harvey, S. Iwata, and V. Mirrokni. Approximating submodular functions everywhere. In *SODA*, pages 535–544, 2009.

[19] H. Ishikawa. Higher-order clique reduction in binary graph cut. In *CVPR*, 2009.

[20] S. Iwata and K. Nagano. Submodular function minimization under covering constraints. In *FOCS*, pages 671–680. IEEE, 2009.

[21] R. Iyer and J. Bilmes. Algorithms for approximate minimization of the difference between submodular functions, with applications. In *UAI*, 2012.

[22] R. Iyer and J. Bilmes. Submodular Optimization with Submodular Cover and Submodular Knapsack Constraints. In *NIPS*, 2013.

- [23] R. Iyer and J. Bilmes. The Lovász-Bregman Divergence and connections to rank aggregation, clustering and web ranking. In *UAI*, 2013.
- [24] R. Iyer, S. Jegelka, and J. Bilmes. Curvature and Optimal Algorithms for Learning and Minimizing Submodular Functions. In *NIPS*, 2013.
- [25] R. Iyer, S. Jegelka, and J. Bilmes. Fast Semidifferential based Submodular function optimization. In *ICML*, 2013.
- [26] R. Iyer, S. Jegelka, and J. Bilmes. Fast Algorithms for Submodular Optimization based on Continuous Relaxations and Rounding: Extended Version, 2014.
- [27] S. Jegelka and J. A. Bilmes. Approximation bounds for inference using cooperative cuts. In *ICML*, 2011.
- [28] S. Jegelka and J. A. Bilmes. Submodularity beyond submodular energies: coupling edges in graph cuts. In *CVPR*, 2011.
- [29] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *SIGKDD*, 2003.
- [30] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE TPAMI*, 26(2):147–159, 2004.
- [31] C. Koufogiannakis and N. Young. Greedy δ -approximation algorithm for covering with arbitrary constraints and submodular cost. *Algorithmica*, 2013.
- [32] A. Krause, A. Singh, and C. Guestrin. Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *JMLR*, 9:235–284, 2008.
- [33] A. Kulik, H. Shachnai, and T. Tamir. Maximizing submodular set functions subject to multiple linear constraints. In *SODA*, 2009.
- [34] H. Lin. *Submodularity in Natural Language Processing: Algorithms and Applications*. PhD thesis, University of Washington, Dept. of EE, 2012.
- [35] H. Lin and J. Bilmes. A class of submodular functions for document summarization. In *ACL*, 2011.
- [36] H. Lin and J. A. Bilmes. Optimal selection of limited vocabulary speech corpora. In *Interspeech*, Florence, Italy, 2011.
- [37] L. Lovász. Submodular functions and convexity. *Mathematical Programming*, 1983.
- [38] M. Minoux. Accelerated greedy algorithms for maximizing submodular set functions. *Optimization Techniques*, pages 234–243, 1978.
- [39] K. Nagano and Y. Kawahara. Structured convex optimization under submodular constraints. In *Proc. UAI*, 2013.
- [40] M. Narasimhan and J. Bilmes. A submodular-supermodular procedure with applications to discriminative structure learning. In *UAI*, 2005.
- [41] G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming*, 14(1):265–294, 1978.
- [42] C. Papadimitriou and K. Steiglitz. *Combinatorial optimization: algorithms and complexity*. Courier Dover Publications, 1998.
- [43] S. Rajagopalan and V. Vazirani. Primal-dual RNC approximation algorithms for set cover and covering integer programs. *SIAM Journal on Computing*, 28(2):525–540, 1998.
- [44] A. Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer Verlag, 2003.
- [45] P. Stobbe and A. Krause. Efficient minimization of decomposable submodular functions. In *NIPS*, 2010.
- [46] P. Stobbe and A. Krause. Learning fourier sparse set functions. In *AISTATS*, 2012.
- [47] Z. Svitkina and L. Fleischer. Submodular approximation: Sampling-based algorithms and lower bounds. In *FOCS*, pages 697–706, 2008.
- [48] S. Vicente, V. Kolmogorov, and C. Rother. Graph cut based image segmentation with connectivity priors. In *Proc. CVPR*, 2008.
- [49] J. Vondrák. *Submodularity in combinatorial optimization*. PhD thesis, Charles University, 2007.
- [50] J. Vondrák. Optimal approximation for the submodular welfare problem in the value oracle model. In *STOC*, pages 67–74. ACM, 2008.
- [51] A. Yuille and A. Rangarajan. The concave-convex procedure. *Neural Computation*, 15(4):915–936, 2003.

Min- d -Occur: Ensuring Future Occurrences in Streaming Sets

Vidit Jain

Yahoo Labs
Bangalore, India

Sainyam Galhotra*

Dept. of Comp. Sci. and Engg.
IIT Delhi, India

Abstract

Given a set of n elements and a corresponding stream of its subsets, we consider the problem of selecting k elements that should appear in at least d such subsets arriving in the “near” future with high probability. For this min- d -occur problem, we present an algorithm that provides a solution with the success probability of at least $1 - O\left(\frac{kd \log n}{D} + \frac{1}{n}\right)$, where D is a known constant. Our empirical observations on two streaming data sets show that this algorithm achieves high precision and recall values. We further present a sliding window adaptation of the proposed algorithm to provide a continuous selection of these elements. In contrast to the existing work on predicting trends based on potential increase in popularity, our work focuses on a setting with provable guarantees.

1 Introduction

Consider a set S_n of n elements. When different sets $S^t \subseteq S_n$ are being observed at time t and may not be analyzed at a later time, we refer to these sets as *streaming sets*. This formulation of streaming sets is ubiquitous at least in the analysis of network traffic (Edwards et al., 1997), query logs (Golbandi et al., 2013), and social media (Mathioudakis and Koudas, 2010), where these sets arrive rapidly. Analyzing these streaming sets to identify historical patterns and predict future trends has been extensively studied for diverse applications including detection of intrusions (Lee et al., 2000), disease outbreak (Achrekar et al., 2011), and viral content (Weng et al., 2013). These works primarily focus on the recall

and the earliness of these predictions. As a result, they are useful for detecting outliers and arranging for preventive measures (Lamb et al., 2013). However, without any lower bound on the obtained precision values, these approaches are ineffective when the false positives may have significant cost associated with them.

Precise predictions are necessary for several planning applications such as resource allocation. For instance, accurate estimates of future traffic may help optimize routing to reduce network latency (Padmanabhan and Mogul, 1996). Similarly, accurate estimates of search query volume may help advertisers optimize marketing budget while bidding for key phrases on a search engine (Jansen and Mullen, 2008). To this end, we study the problem of making these predictions only when we can guarantee a minimum probability of success before the time horizon Δ . We define this problem formally as follows.

Definition 1. Min- d -occur. *Given a stream of sets $S^t \subseteq S_n$ arriving at time t , min- d -occur(S_n, Δ, k) selects at most k elements $\{y_1, y_2, \dots, y_k\} \in S_n$ at time τ such that each of these y_j appears at least d number of times in $\{S^{\tau+1}, \dots, S^\Delta\}$ with probability close to 1.*

Note that a solution to this problem may select fewer than k elements when the criterion for the probabilistic guarantee is not met. The constant Δ depends on the requirements of the application domain, which can be understood through an illustrative example as follows. Consider a stream of queries issued to a search engine that may be arriving at a rate of one million queries per second. Choosing Δ close to 100K would correspond to making the predictions about occurrences in the next 100 milliseconds. The choice of the parameter d depends on the characteristics of the stream of sets and determines the effectiveness of the solution. For the stream of queries, let us assume that we wish to track a set of 50 popular queries (i.e., $n = 50$), which cover approximately 1% of the incoming stream. In other words, 99% of the stream would be composed of empty sets, making the solution likely to be effective

*Work done during an internship at Yahoo Labs Bangalore.

only when $d \leq 1\%$ of Δ . Since these domain-specific stream characteristics are external to our problem formulation, we ignore the empty sets hereafter.

In the absence of any other assumptions about the stream characteristics, the min- d -occur problem is ill-posed. In fact, if there is a sudden drop in the frequency of the queries from the set S_n in the query stream (in the above example), there may not exist a solution set for an algorithm to predict. Using an appropriate selection of the set S_n and an assumption about the continuity of the stream statistics in the given time horizon Δ , it might be fair to assume that at least k queries will appear at least $D = 100$ times in the next 100 milliseconds. Under this additional assumption, the above min- d -occur problem becomes more useful for $d \leq D$. Using this constant integer D , we define a constrained variant as follows.

Definition 2. Constrained-min- d -occur. *Given that there exist at least k elements in S_n that appear D number of times in $\{S^1, \dots, S^\Delta\}$, constrained-min- d -occur(S_n, D, Δ, k) solves min- d -occur(S_n, Δ, k).*

Below we present a randomized algorithm for the constrained-min- d -occur problem. We show that the probability of a successful prediction from this algorithm is high when $d < \frac{D}{3 \log n}$ and $k = o(n/\log n)$. This theoretical result is further validated using experiments on two streaming data sets: search query logs and hash-tags in tweets. In both of these data sets, this algorithm achieved high precision and recall values for predictions. Interestingly, this algorithm is empirically effective for larger values of d even when a corresponding lower-bound is not computed. We also present a sliding-window based adaptation of our algorithm to accommodate a practical setting where the predictions need to be updated only at regular intervals of time.

Contributions. There are two main theoretical contributions in this paper. First, we present a novel problem formulation – *constrained min- d -occur* – for making guaranteed predictions of future occurrences in streaming data. Second, we present a randomized algorithm for making predictions under this formulation and derive an effective lower-bound on its performance. These theoretical contributions are supported by empirical observations on two real, streaming data sets i.e., Twitter hash-tags and query logs of a search engine.

2 Related Work

Our problem formulation is related to online coverage problems. In the online set cover problem, Alon et al. (2009) assume that a collection of sets is known be-

forehand but they arrive in an online fashion. An algorithm then selects k sets from this collection that solves the max-cover problem for the sets that are already observed. Whereas our setup requires predictions to be made for the sets arriving in future. Another related coverage problem is set-streaming maximum coverage proposed by Saha and Getoor (2009). They assume an orthogonal set up where the collection of sets remain static but their elements are streaming. Their algorithm is not applicable in our setting because we need to choose the elements that are present in maximum number of sets, and *not* the sets that cover maximum number of elements. Another similar problem formulation has appeared in the operating systems literature. Awerbuch et al. (1996) studied the problem of allocating jobs to workstations to ensure a timely, successful completion of the given jobs. They presented an algorithm that performs a sequential prediction of these allocations and allows at most one job to be running at any time. That is, the second job is allocated only after the completion (*not* allocation) of the first job. It is non-trivial to adapt their algorithm to perform k simultaneous allocations, although we have borrowed useful concepts from their lower-bound analysis in the development of our solution.

There exists significant literature on statistical trend prediction in streaming data, most notably for the Twitter data. There has also been some work on tracking topics and events in these streams (Ardon et al., 2013) through an appropriate adaptation of statistical topic models (AlSumait et al., 2008), and for detecting bursts or spikes in topics (Diao et al., 2012). Goorha and Ungar (2010) study the spiking behavior of elements in an input set of elements. Their algorithm can only handle a small set of elements, thereby limiting its utility in a general practical setting. Cataldi et al. (2010) modeled the streaming elements as a graph, and employed page-rank algorithm along with an aging theory to predict the spiking elements. Mathioudakis and Koudas (2010) analyzed the sudden change in frequency patterns of these elements in conjunction with a reputation model for the origin of the streaming elements to make these predictions. Becker et al. (2011) discussed an online setting to identify events and the related tweets, but they did not make predictions for future. Similarly, there has also some work on analyzing trends in user comments (Jain and Galbrun, 2013)

Most of these algorithms are optimized for a specific application, and require significant modification to be applicable to a different setting. These algorithms are evaluated empirically using measures such as the perplexity, recall, and earliness of the predictions. Since these approaches do not formally specify the (implicit) assumptions made about the stream character-

S_n	set of elements $\{e_1, \dots, e_n\}$
n	number of possible elements in the global set
k	number of elements to choose
x_e^t	Score of the element e at time t
y^t	Solution set constructed at time t
N	window size
w_e^t	window data structure for element e at time t

Table 1: Notation used in this paper

istics, it is challenging to assess their utility across different related settings, e.g., our current formulation. Also, lower-bound analyses for the prediction accuracy even for the original problem settings do not exist. In this work, we address both of these issues. We formally stated the assumptions of our prediction setting in the previous section. In the next section, we present a randomized algorithm to obtain a solution for this setting. A lower-bound analysis of the performance of this algorithm is described in the subsequent section.

3 Algorithm

Here we present a randomized algorithm that provides the constrained-min- d -occur (as defined above). In other words, this algorithm selects k elements (from S_n) that are likely to appear at least d number of times in the given future Δ occurrences. As we show below, the probability of a successful prediction from this algorithm is high when: $D > 3d \log n$ and $k = o(n/\log n)$. Later, in Section 6, we demonstrate that these assumptions hold in several practical settings for streaming data.

Algorithm 1 describes the different steps of our approach. We maintain a score x_e^t for each element $e \in S_n$ at every time t . At time t , a new set S^t is presented to the algorithm. For each element e present in the S^t that has already not been selected, we toss a coin with probability of getting heads $= \frac{3x_e^t - 2}{d}$. If a head is observed, we add e to the selection set y . We update the score x_e^t and continue till k elements have been selected. The set y represents the selected k elements.

4 Analysis

Here we prove that each of the k elements $\{y_1, y_2, \dots, y_k\}$ selected by our algorithm has a high probability of occurring at least d times in future (governed by the horizon Δ). We show this probability to be greater than $1 - O(\frac{k d \log n}{D})$.

We approach this proof by constructing a subspace

Algorithm 1 k Min- d -occur Prediction

Require: set of elements S_n , integer k , stream $\{S^1, \dots, S^\Delta\}$

- 1: $n \leftarrow |S_n|$.
- 2: $\forall e \in S_n, x_e^0 \leftarrow 0$.
- 3: $y \leftarrow \{\phi\}$.
- 4: $j = 1$
- 5: **for** $t = 1$ to Δ **do**
- 6: $\beta^t \leftarrow S^t \setminus y$
- 7: **for** $e \in \beta^t$ **do**
- 8: $x_e^t \leftarrow x_e^{t-1} + 1$.
- 9: choose $u \sim \text{Bernoulli}\left(\frac{n(3x_e^t - 2D)/D}{d}\right)$.
- 10: **if** $u = 1$ **then**
- 11: $y_j \leftarrow e$.
- 12: $j \leftarrow j + 1$.
- 13: **if** $j > k$ **then**
- 14: **return** y .
- 15: **end if**
- 16: **end if**
- 17: **end for**
- 18: **end for**

S_u of probabilistic outcomes of coin-toss experiments (as illustrated in Figure 1). In this subspace, 0 denotes the absence of an element in the set arriving at a given time. Whereas the presence of a particular element is denoted by the probabilistic outcome, i.e., the toss of the coin as H (head) or T (tail). In other words, there will be one coin toss for each pair (t, i) if and only if $e_i \in S^t \setminus y^t$. The element e_{i^*} is selected as the j^{th} prediction at time t^* if and only if:

- (t^*, i^*) toss is H ,
- for $t < t^*$ and $e_i \notin y^t$, (t, i) toss is T ,
- for $t = t^*$ and $i < i^*$, (t, i) toss is T .

Let $S_o \subseteq S_u$ be the subspace where each of the k selected elements occur in d sets after choosing them. We refer to this subspace as the solution space. In this subspace, one H appears in each of the k different values of i (in different columns) and there are at least d such sets. Each element would appear in at least d sets after their respective coin tosses are heads.

To show that $Pr[S_o]$ is close to 1, we first choose an intermediate subspace $S_i \subseteq S$ for which the corresponding probability, i.e., $Pr[S_i]$, is close to 1. This intermediate subspace is selected such that we can construct a useful injection from S_i to S_o . To this end, we define S_i to consist of sample points for which there are at least k heads for different elements and for which the first d flips for each element are tails.

	e_1	e_2	e_3	e_4	e_5	e_6	e_7
α^1	0	0	(1,T)	(1,T)	0	0	0
α^2	0	(1,T)	0	(1,T)	(1,T)	0	0
\cdot	\cdot	0	\cdot	(1,H)	0	(1,H)	0
\cdot	\cdot	\cdot	\cdot	\cdot	0	\cdot	\cdot
\cdot	\cdot	\cdot	\cdot	\cdot	\cdot	\cdot	\cdot
\cdot	\cdot	\cdot	\cdot	\cdot	\cdot	\cdot	\cdot
\cdot	\cdot	\cdot	\cdot	\cdot	\cdot	\cdot	\cdot
\cdot	\cdot	\cdot	\cdot	\cdot	\cdot	\cdot	\cdot
\cdot	\cdot	\cdot	\cdot	\cdot	\cdot	\cdot	\cdot
α^i	(1,T)	(1,T)	(1,H)	(1,H)	(1,T)	(1,H)	(1,T)

(a) Universal space S_u

	e_2	e_3	e_4	e_5	e_6
α^1	0	(1,T)	(1,T)	0	0
α^2	(1,T)	0	(1,T)	(1,T)	(1,T)
\cdot	\cdot	\cdot	\cdot	0	(1,T)
\cdot	\cdot	\cdot	\cdot	0	\cdot
\cdot	(1,T)	\cdot	\cdot	\cdot	(1,T)
\cdot	\cdot	(1,T)	(1,T)	\cdot	\cdot
\cdot	\cdot	\cdot	\cdot	\cdot	\cdot
\cdot	\cdot	\cdot	\cdot	\cdot	\cdot
\cdot	\cdot	\cdot	\cdot	\cdot	\cdot
\cdot	\cdot	\cdot	\cdot	(1,T)	\cdot
α^i	(1,H)	(1,H)	(1,H)	(1,T)	(1,H)

(b) Intermediate space S_i

	e_2	e_3	e_4	e_5	e_6
α^1	0	(1,T)	(1,T)	0	0
α^2	(1,T)	0	(1,T)	(1,T)	(1,T)
\cdot	\cdot	\cdot	\cdot	0	(1,T)
\cdot	\cdot	\cdot	\cdot	0	\cdot
\cdot	(1,H)	\cdot	\cdot	\cdot	(1,H)
\cdot	\cdot	(1,H)	(1,H)	\cdot	\cdot
\cdot	\cdot	\cdot	\cdot	\cdot	\cdot
\cdot	\cdot	\cdot	\cdot	\cdot	\cdot
\cdot	\cdot	\cdot	\cdot	\cdot	\cdot
\cdot	\cdot	\cdot	\cdot	(1,T)	\cdot
α^i	(1,H)	(1,H)	(1,T)	(1,T)	(1,T)

(c) Solution space S_o

Figure 1: *Illustration of the probabilistic subspace for $n = 7$.* Each row represents a set α , where the absence of an element is denoted by 0; the outcome of the coin-toss for each of the other elements is shown as a tuple $(1, H)$ or $(1, T)$.

From the columns of S_u , only those elements are kept for which the first d tosses are tails to form the subspace S_i . The mapping from S_i to S_o has also been shown where the $(x_e^t - d)^{th}$ and x_e^t outcomes of coin toss are flipped depending upon the factors explained later. We start by proving a basic result about the universal space S_u .

Lemma 1. *The probability of getting tails for all coin tosses in $S_u \leq e^{-n/2}$.*

Proof. The probability of getting tails for all elements is the product of probability of getting tails for each element. The latter probability is less than the probability of getting tails for an element that is present in at least D sets. This upper bound probability is the same as the probability that there are no heads among the last d flips for the element that is available for D steps, which is at most

$$\left(1 - \frac{n^{\frac{3(D-d)}{D}-2}}{d}\right)^d \leq \left(1 - \frac{n}{2d}\right)^d \leq e^{-n/2} \quad (1)$$

because $D \geq 3d \log n$. \square

The probabilistic subspace S_i consists of observation sets such that there are at least k heads for different elements and for which the first d flips for each element are tails. Now we lower bound the probability associated with this subspace.

Lemma 2. $Pr[S_i] \geq 1 - O(1/n)$.

Proof. We found the probability of the complement of S_i , denoted by \bar{S}_i , to be easier to compute than for the subspace S_i . This probability is decomposed as a sum of two terms P_1 and P_2 defined below. The term P_1 denotes the probability of getting a head among the first (at most) $d \times n$ flips for each $x_i^t \leq d$. Since the

probability of observing an H is at most $\frac{n^{\frac{3d}{D}-2}}{d}$,

$$P_1 \leq dn \left(\frac{n^{\frac{3d}{D}-2}}{d} \right) \leq 2/n. \quad (2)$$

Let P_2 denote the probability of observing at most $k-1$ heads, i.e.,

$$P_2 = \sum_{i=0}^{k-1} Pr_i[H], \quad (3)$$

where $Pr_i[H]$ is the probability of observing a total i occurrences of H . Assuming i.i.d. observations for coin-tosses, we have

$$P_2 = \sum_{i=0}^{k-1} \binom{n}{i} Pr_1[H]^i Pr_1[T]^{n-i} \quad (4)$$

$$\leq \sum_{i=0}^{k-1} \binom{n}{i} Pr_1[T]^{n-i}, \quad (5)$$

since $Pr_1[H] \leq 1$. Keeping only those elements for which we get tails and they appear in at least D sets, we have

$$P_2 \leq \sum_{i=0}^{k-1} \binom{n}{i} Pr_1[T]^{k-i} \quad (6)$$

$$\leq k \times \max \left(\binom{n}{i} \times (e^{-n/2})^{k-i} \right). \quad (7)$$

For $k < n/2$, $\binom{n}{i} \times (e^{-n/2})^{k-i}$ is an increasing function with respect to i ; the maximum value is obtained for $i = k-1$. Therefore

$$P_2 \leq k \binom{n}{k-1} \times (e^{-n/2})^{k-(k-1)} \quad (8)$$

$$= k \left(\binom{n}{k-1} \times e^{-n/2} \right). \quad (9)$$

Also, since $\binom{n}{k} \leq n^k$,

$$P_2 \leq k \times n^{k-1} e^{-n/2} \leq 1/n, \quad (10)$$

if $k \leq \left(\frac{n}{2^{\log n}} - 1\right)$. Finally, combining Equation 2 and 10

$$Pr[S_i] = 1 - (P_1 + P_2) \quad (11)$$

$$= 1 - (O(2/n) + O(1/n)) \quad (12)$$

$$= 1 - O(1/n). \quad (13)$$

□

For each of the members of the above intermediate subspace, we construct a member of the solution space. We ensure that the probabilities of occurrences of both of these instances are similar. The next lemma provides the similarity in the probabilities of these two subspaces.

Lemma 3. $Pr[S_o] \geq (1 - O(\frac{kd \log n}{D})) Pr[S_i]$.

Proof. We construct an injection $f : S_i \rightarrow S_o$ such that $\forall s' \in S_i$,

$$Pr[f(s')] \geq \left(1 - O\left(\frac{kd \log n}{D}\right)\right) Pr[s']. \quad (14)$$

Consider an instance $s' \in S_i$. Let $\{e_{i_1}, e_{i_2}, \dots, e_{i_k}\}$ be the elements for which the flips are the first heads in s' . Let x_{i_j} denote the number of sets in which e_{i_j} has been present and includes the set where the j^{th} heads occurred for $j \in \{1, 2, \dots, k\}$. By definition of S_i , $x_{i_j} > d$. Let us define

$$z'_{i_j} = n^{\frac{3x_{i_j}}{D} - 2} / d \quad (15)$$

and

$$z_{i_j} = \frac{n^{\frac{3(x_{i_j} - d)}{D} - 2}}{d} = n^{\frac{-3d}{D}} z'_{i_j} \quad (16)$$

$$= \left(1 - O\left(\frac{d \log n}{D}\right)\right) z'_{i_j}. \quad (17)$$

Let p refers to the sets considered in a sample space. Using these notation, we define the injection $f(s')$ as follows. If $z'_{i_j} \geq 1/2$ then all of the (p, i_j) pairs of $f(s')$ are made identical to the respective (p, i_j) of s' . Next, the $(x_{i_j} - d)^{th}$ flip of e_{i_j} is changed from tails to heads.

If $z'_{i_j} \leq 1/2$ then all of the (p, i_j) pairs of $f(s')$ are similar to (p, i_j) of s' except that the $(x_{i_j} - d)^{th}$ flip of e_{i_j} is changed from tails to heads and $x_{i_j}^{th}$ flip of e_{i_j} is changed from heads to tails.

The above process generates $f(s') \in S_o$. Without loss of generality, let us assume that for $j \in \{1, 2, \dots, r\}$ the value of $z'_{i_j} \geq 1/2$. For $j \in \{r+1, r+2, \dots, k\}$, let us assume $z'_{i_j} \leq 1/2$ where r is an integer $1 \leq r \leq k$.

Using this notation, the ratios of probabilities in the two subspaces is given by:

$$\frac{Pr[f(s')]}{Pr[s']} = \prod_{i_j=1}^r \frac{z_{i_j}}{1 - z_{i_j}} \cdot \prod_{i_j=r+1}^k \frac{z_{i_j}}{1 - z_{i_j}} \frac{1 - z'_{i_j}}{z'_{i_j}} \quad (18)$$

For $z'_{i_j} \geq 1/2$, using Equation 17, we get

$$\frac{z_{i_j}}{1 - z_{i_j}} = \frac{\left(1 - O\left(\frac{d \log n}{D}\right)\right) z'_{i_j}}{1 - \left(1 - O\left(\frac{d \log n}{D}\right)\right) z'_{i_j}} \quad (19)$$

$$\geq \frac{1/2 - O\left(\frac{d \log n}{D}\right)}{1/2 + O\left(\frac{d \log n}{D}\right)} \quad (20)$$

$$= 1 - O\left(\frac{d \log n}{D}\right). \quad (21)$$

For $z'_{i_j} \leq 1/2$, using Equation 17, we get

$$\left(\frac{z_{i_j}}{1 - z_{i_j}}\right) \left(\frac{1 - z'_{i_j}}{z'_{i_j}}\right) \quad (22)$$

$$\geq \left(1 - O\left(\frac{d \log n}{D}\right)\right) \frac{1 - z'_{i_j}}{1 - z'_{i_j} + O\left(\frac{d \log n}{D} z'_{i_j}\right)}$$

$$\geq 1 - O\left(\frac{d \log n}{D}\right). \quad (23)$$

Using Equation 21 and Equation 23, the Equation 18 reduces to the following.

$$\frac{Pr[f(s')]}{Pr[s']} \geq \prod_{i_j=1}^k \left(1 - O\left(\frac{d \log n}{D}\right)\right) \quad (24)$$

$$\geq 1 - O\left(\frac{kd \log n}{D}\right). \quad (25)$$

□

Theorem 3. $Pr[S_o] \geq 1 - O\left(\frac{kd \log n}{D} + \frac{1}{n}\right)$.

Proof. Using Lemma 2 and Lemma 3. □

Corollary 4. For $k = 1$, we obtain a $2/3$ approximation, i.e., $Pr[S_o] \geq 2/3$ since $D > 3d \log n$.

5 Sliding Window Approach

In a practical setting, the predictions about future occurrences are made continuously as the streaming sets continue to arrive. Alternatively, the predictions are updated at regular intervals of time using the sets arriving within a sliding time window of size W . The algorithm described above would require a re-computation of the frequency counts of the elements

as the window is updated. To reduce the computational cost of this update, we employ a data structure proposed by Datar et al. (2002) that maintains count statistics for a stream using buckets of different sizes. Using the binary representation for the presence of an element in a streaming set, we use this data structure to calculate the approximate number of 1's in a given window. For each of these buckets, the time-stamp of the most recent 1 and the total count of 1's appearing in that bucket are maintained. As a new binary element arrives the following steps are taken.

- The time stamp of each bucket is increased by one.
- If the time stamp of a bucket exceeds the window size, the bucket is dropped.
- If the arriving element is 0, we proceed to the next element.
- If the arriving element is 1, then we create a new bucket with size 1 (this operation is referred to as `add_bucket` operation).
- if there exists $m/2 + 2$ buckets of same size, the oldest two buckets are merged. Here m is a pre-defined integer that is inversely proportional to the maximum permissible relative error ϵ , i.e., $m = \lceil 1/\epsilon \rceil$.

In this data structure, the total number of elements in each bucket is equal to the number of 1's in the window (referred to as `window_score`). Datar et al. (2002) show that this data structure gives an estimate of the number of 1's in the window of size W with a relative error of at most ϵ using at most $(\frac{m}{2} + 1) (\log(\frac{2W}{m} + 1) + 1)$ buckets. $\log W + \log \log W$ bits are used per bucket and each new element is processed in $O(\log W)$ worst case time. At each instant, this data structure provides a count estimate in $O(1)$ time.

Algorithm 2 use `add_bucket` and `window_score` to present the sliding window adaptation of Algorithm 1. In our problem formulation, sets $\{S^1, S^2, \dots, S^W\}$ arrive in the window W . Let there be x elements in the union of these sets, then the input stream for the above data structure would be a permutation of x 1's and $(n - x)$ 0's. For this stream, we create n buckets at the beginning of the stream and update them as above when a new element arrives.

6 Experiments

The above analysis ensures a high probability of successful predictions from our algorithm. To understand

Algorithm 2 k Min-d-occur Prediction over a sliding window

Require: S_n , integer k , window size W .

```

1:  $\forall e \in S_n, x_e^0 \leftarrow 0$ .
2:  $j = 1$ 
3: for  $t = 1$  to  $\Delta$  do
4:    $\beta^t \leftarrow S^t \setminus y$ 
5:   for  $e \in \beta^t$  do
6:      $w_e^t \leftarrow \text{add\_bucket}(w_e^{t-1})$ 
7:      $x_e^t \leftarrow \text{window\_score}(x_e^{t-1}, w_e^t)$ .
8:     choose  $u \sim \text{Bernoulli}\left(\frac{n(3x_e^t - 2D)/D}{d}\right)$ .
9:     if  $u = 1$  then
10:       $y_j \leftarrow e$ .
11:       $j \leftarrow j + 1$ .
12:      if  $j > k$  then
13:        return  $y$ .
14:      end if
15:    end if
16:  end for
17: end for
```

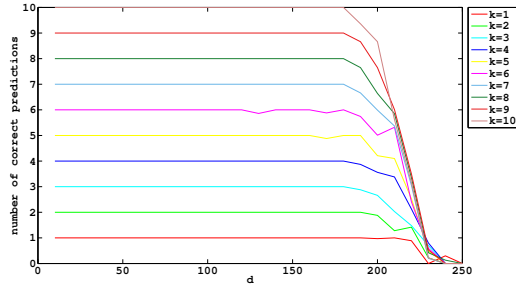
its effectiveness in practical settings, we performed experiments on two real-world collections of streaming data: query logs from a commercial search engine and hash-tags appearing in tweets.

A naïve approach based on selecting the most frequent elements could be considered as a baseline approach. However, such comparisons would be unfair to the baseline approaches because they would not be directly optimizing the criterion our problem setup mandates. More importantly, such comparisons would risk the clarity in the distinction of the proposed problem formulation and the traditional setup for statistical trend prediction. That said, the results for a naïve approach were indeed empirically inferior to ours.

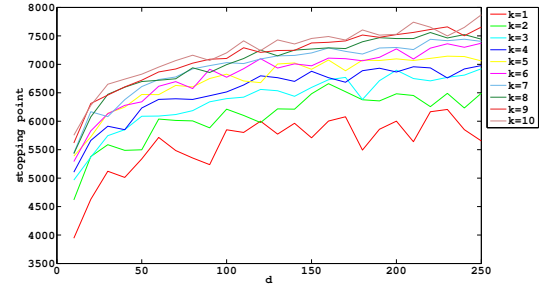
6.1 Search queries

We considered a data set comprising of one month of anonymized search logs that is made public under the Yahoo Webscope program¹. We only consider the anonymized query identifiers in our experiments, ignoring other information – i.e., document identifiers, relevance judgments, etc. – present in this data set. We randomly sample 100K queries and use their respective timestamp to simulate the arrival of these queries as a stream. Each of these queries corresponds to a singleton, streaming set in our formulation. Figure 3 shows the frequency distribution of the resulting 603 unique queries in this data set. We select all of these unique queries to construct the set S_n (i.e.,

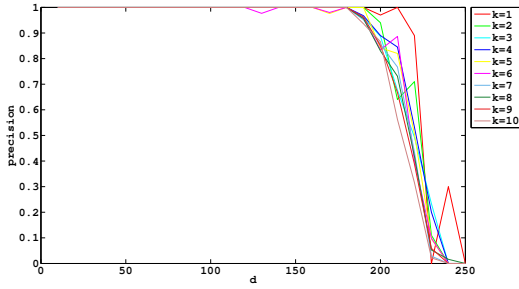
¹The L18 data set available from <http://webscope.sandbox.yahoo.com>



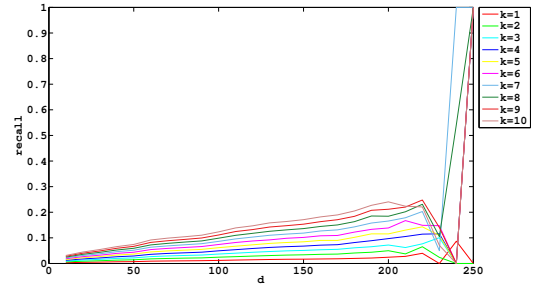
(a) Average Number of correct predictions



(b) Number of streaming subsets observed before making a prediction



(c) Precision



(d) Recall

Figure 2: *Results on search queries.* Each of the four plots show different evaluation metrics for different values of k and d and $D = 200$. These metrics are averaged over 200 iterations of our algorithm on random sub-streams of the original data stream. (Best seen in color).

$n = 603$). There are 140 queries that appear at least 200 times in the entire stream, therefore the assumption required for the constrained-min- d -occur problem clearly holds for $D = 200$ and $k \leq 10$. We ran our algorithm for different values of d and report our observations in Figure 2. Even though we do not have a valid lower-bound on the prediction accuracy for $d > D$, we examine if the algorithm is still able to make predict queries with larger number of future occurrences.

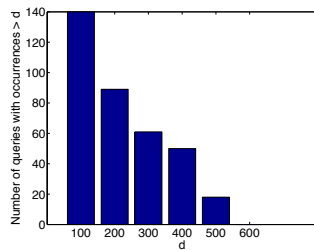


Figure 3: *Frequency distribution of search queries.*

The precision and recall metrics shown in these plots are computed by comparing the set of predicted queries against the queries that actually occur d times in the stream remaining after the prediction is made. For $d = 150$, the predicted queries were 100% accu-

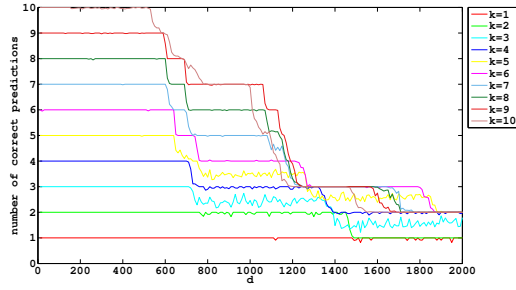
rate for $k \leq 10$. On average, the algorithm was able to make predictions for $k = 10$ after observing about 7500 queries. In the remaining stream, about 60 unique queries appeared more than 150 times. Therefore, the recall of our algorithm is around 0.16.

As we increase d , the probability of observing heads in a coin-toss decreases (see Section 4), making the algorithm take longer to make the predictions. Outside the provably correct range $d < \frac{D}{3 \log n}$, the precision is also observed to be adversely affected in a similar manner. For larger values of k , the algorithm takes more time to make predictions, and the performance degrades outside the provably correct range. In the provably correct range, the recall values for $k = 2$ is nearly twice the value for $k = 1$, because the number of predictions made are twice in the former case and all of the predictions are accurate; the minor deviation is attributed to the difference in the stopping point in the two experiments.

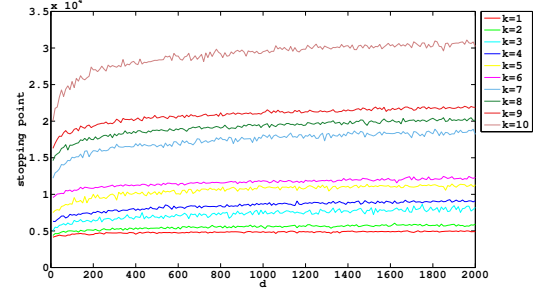
6.2 Twitter hash tags

We used the Twitter’s public REST API² to obtain 150K tweets from the month of November 2013. We

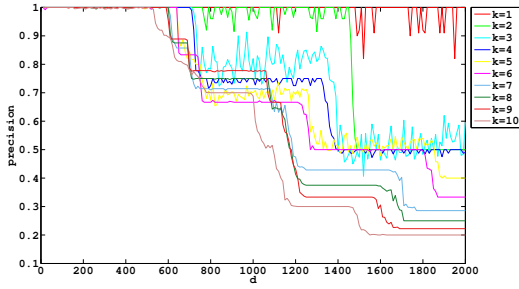
²<https://dev.twitter.com/docs/api/1.1>



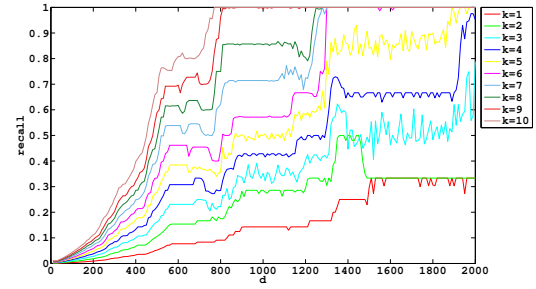
(a) Average Number of correct predictions



(b) Number of streaming subsets observed before making a prediction

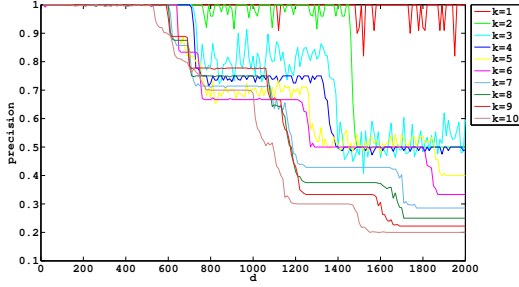


(c) Precision

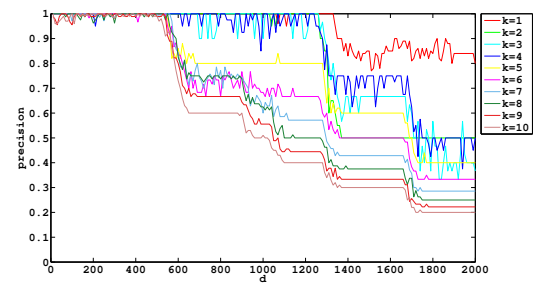


(d) Recall

Figure 5: *Results on Twitter hash-tags.* Each of the four plots show different evaluation metrics for different values of k and d and $D = 200$. These metrics are averaged over 200 iterations of our algorithm on random sub-streams of the original data stream. (Best seen in color).



(a) $D=200$



(b) $D=400$

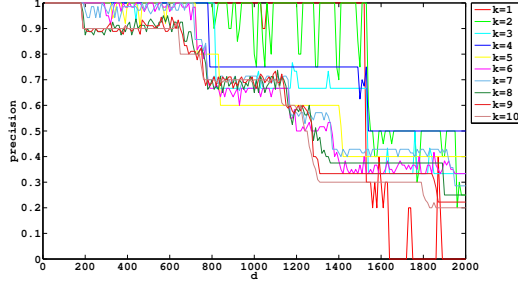
Figure 6: *Comparison of precision curves for different choices of D on Twitter hash-tags.*

consider all of the hash-tags appearing in each of these tweets to compose the respective streaming set. We selected the 1K most frequent hash-tags in this collection to construct S_n . Figure 4 shows the frequency distribution of the unique hash-tags in this data set. Considering the skew in this distribution, we selected a larger value of $D = 200$ for this data set.

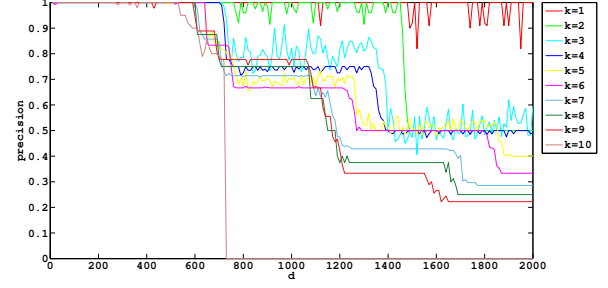
The overall observations for this data sets are similar to those for the previous data set. However, the drop in performance is gradual over a bigger range of values of d . For $k = 1$, the algorithm continued to make almost

accurate predictions till $d = 10 * D = 2000$. For $k = 10$, around 70% precision was observed while predicting 1000 future occurrences. The algorithm only observed 30K tweets to make these predictions. Assuming an estimated arrival rate of 150K tweets per minute, our algorithm can start predicting these ten tags with 70% accuracy in less than a minute.

The parameter D has a trade-off associated with it. On one hand, a larger value of D reduces the time taken to make a prediction and increases the provably useful range of values of d . On the other hand,



(a) $D=100$



(b) $D=200$

Figure 7: Results for the sliding-window approach on Twitter hash-tags. The window size is set of $20K$. The average number of correct predictions(Accuracy) made for $D = 100$ and 200 for different values of k and d .

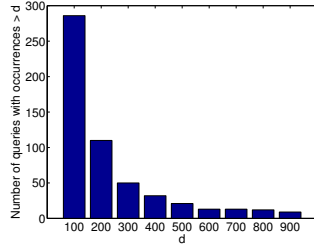


Figure 4: Frequency distribution of unique hash-tags

we may have to increase the size of the stream to ensure that the related assumption for constrained-min- d -occur holds. Figure 6 shows a comparison of the observed precision values for two different choices of $D = \{200, 400\}$. In both of these plots for most values of k , the precision is close to 1 for almost identical range of d , but the precision drops faster for smaller value of D . Also, the graphs for $D = 400$ are noisy, which suggests a large variance to be associated with their precision estimates. For this reason, we postulate that useful conclusions could not be derived for $D > 400$ in this data set.

6.3 Sliding window

Figure 7 shows the precision curves obtained using the sliding window adaptation of our algorithm (described in Section 5) with window size $W = 20K$. For $D = 200$, the graphs validate two hypothesis about the correctness of our window-based algorithm. First, for $k = 10$, the stopping point (see Figure 5) is greater than the window size. Therefore, the window-based algorithm should fail to make predictions for most values of d . Second, the stopping points for $k < 10$ are less than the window size for almost all of the values of d . Hence, the window-based algorithm should obtain similar performance as the original algorithm without the window. Similar observations were made when a

window size $W = 10K$ was considered. Similar observations were made for $D = 100$. The relative difference between the performances for $D = 100$ and $D = 200$ can be explained similar to the discussion for Figure 6. These observations confirm that sliding window adaptation of the original algorithm well approximates the desired solution. This approximate algorithm is better suited for a practical setting where predictions need to be made at regular intervals of time.

7 Discussion

We presented a new problem formulation, *constrained min- d -occur*, for studying algorithms that guarantee a minimum number of future occurrences in streaming data. For this formulation, we presented a randomized algorithm and derive a lower-bound on the probability of successful predictions obtained from this algorithm. To our knowledge, any prior theoretical results for this problem does not exist. The theoretical result is further validated using experiments on two real-world data sets: search query logs and hash-tags in tweets. In both of these data sets, the proposed algorithm achieved high precision and recall values for predictions. We studied the performance of this algorithm for different choices of parameters. We also presented a sliding-window based adaptation of our algorithm to accommodate a practical setting where the predictions need to be updated only at regular intervals of time. Interestingly, these algorithms were found to be effective for larger values of d for which a useful lower-bound is not computed. Studying the tightness of our current lower-bound would be a useful extension of this work.

References

Harshavardhan Achrekar, Avinash Gandhe, Ross Lazarus, Ssu-Hsin Yu, and Benyuan Liu. Predicting flu trends using twitter data. 2011.

- Noga Alon, Baruch Awerbuch, Yossi Azar, Niv Buchbinder, and Joseph Seffi Naor. The online set cover problem. *SIAM J. Comput.*, 39(2):361–370, June 2009.
- Loulwah AlSumait, Daniel Barbará, and Carlotta Domeniconi. On-line lda: Adaptive topic models for mining text streams with applications to topic detection and tracking. In *ICDM*, pages 3–12, 2008.
- Sebastien Ardon, Amitabha Bagchi, and Anirban et al. Mahanti. Spatio-temporal and events based analysis of topic popularity in twitter. In *CIKM*, pages 219–228. ACM, 2013.
- Baruch Awerbuch, Yossi Azar, Amos Fiat, and Tom Leighton. Making commitments in the face of uncertainty: how to pick a winner almost every time (extended abstract). In *STOC*, pages 519–530. ACM, 1996.
- Hila Becker, Feiyang Chen, Dan Iter, Mor Naaman, and Luis Gravano. Automatic identification and presentation of twitter content for planned events. In *ICWSM*, 2011.
- Mario Cataldi, Luigi Di Caro, and Claudio Schifanella. Emerging topic detection on twitter based on temporal and social terms evaluation. In *MDMKDD*, 2010.
- Mayur Datar, Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Maintaining stream statistics over sliding windows. In *SODA*, pages 635–644, 2002.
- Qiming Diao, Jing Jiang, Feida Zhu, and Ee-Peng Lim. Finding bursty topics from microblogs. In *ACL*, pages 536–544, 2012.
- T. Edwards, D. S. W. Tansley, R. J. Frank, N. Davey, and Northern Telecom (nortel Limited). Traffic trends analysis using neural networks. In *Proceedings of the International Workshop on Applications of Neural Networks to Telecommunications*, pages 157–164, 1997.
- Nadav Golbandi, Liran Katzir, Yehuda Koren, and Ronny Lempel. Expediting search trend detection via prediction of query counts. In *WSDM*, pages 295–304. ACM, 2013.
- Saurabh Goorha and Lyle Ungar. Discovery of significant emerging trends. In *KDD*, pages 57–64. ACM, 2010.
- Vidit Jain and Esther Galbrun. Topical organization of user comments and application to content recommendation. In *WWW*, 2013.
- Bernard J. Jansen and Tracy Mullen. Sponsored search: an overview of the concept, history, and technology. *IJEB*, 6(2):114–131, 2008.
- Alex Lamb, Michael J. Paul, and Mark Dredze. Separating fact from fear: Tracking flu infections on twitter. In *HLT-NAACL*, pages 789–795, 2013.
- Wenke Lee, Salvatore J. Stolfo, and Kui W. Mok. Adaptive intrusion detection: a data mining approach. *Artificial Intelligence Review*, 14:533–567, 2000.
- Michael Mathioudakis and Nick Koudas. Twittermonitor: trend detection over the twitter stream. In *SIGMOD*, pages 1155–1158. ACM, 2010.
- Venkata N. Padmanabhan and Jeffrey C. Mogul. Using predictive prefetching to improve world wide web latency. *Computer Communication Review*, 26:22–36, 1996.
- Barna Saha and Lise Getoor. On maximum coverage in the streaming model & application to multi-topic blog-watch. In *SDM’09*, pages 697–708, 2009.
- L. Weng, F. Menczer, and Y.-Y. Ahn. Virality prediction and community structure in social networks. *Sci. Rep.*, 3(2522), 2013.

Instance Label Prediction by Dirichlet Process Multiple Instance Learning

Melih Kandemir
Heidelberg University HCI/IWR
Germany

Fred A. Hamprecht
Heidelberg University HCI/IWR
Germany

Abstract

We propose a generative Bayesian model that predicts instance labels from weak (bag-level) supervision. We solve this problem by simultaneously modeling class distributions by Gaussian mixture models and inferring the class labels of positive bag instances that satisfy the multiple instance constraints. We employ Dirichlet process priors on mixture weights to automate model selection, and efficiently infer model parameters and positive bag instances by a constrained variational Bayes procedure. Our method improves on the state-of-the-art of instance classification from weak supervision on 20 benchmark text categorization data sets and one histopathology cancer diagnosis data set.

1 INTRODUCTION

Automated data acquisition has reached unprecedented scales. However, annotation of ground-truth labels is still manual in many applications, lagging behind the massive increase in observed data. This fact makes learning from partially labeled data emerge as a key problem in machine learning. Multiple instance learning (MIL) tackles this problem by learning from labels available only for instance groups, called *bags* [7]. A negatively labeled bag indicates that all instances have negative labels. In a positively labeled bag, there is at least one positively labeled instance; however, which of the instances are positive is not specified. We refer to these bag labeling rules as *multiple instance constraints*. A positive bag instance with a positive label is called a *witness*, and one with a negative label a *non-witness*.

The classical MIL setup involves both bag-level training and bag-level prediction. The mainstream MIL algorithms are developed and evaluated under this classical setup. The harder problem of *instance-level* prediction from *bag-level*

training has been addressed in a comparatively smaller volume of studies [16, 17, 32]. A group of existing models, such as Key Instance SVM (KI-SVM) [16] and CkNN-ROI [32] aim to identify a single positive instance from each positive bag, the so called *key instance*, that determines the bag label, and discard the other instances. In a recent work, Liu et al. [17] generalize this approach by a voting framework (VF) that learns an arbitrary number of key instances from each positive bag. While KI-SVM extends the MI-SVM formulation [2] with binary variables indicating key instances, CkNN-ROI and VF are built on the Citation k-NN method [26].

1.1 Contribution

Our central assumption is that all instances belonging to the same Gaussian / cluster share the same class label. By performing simultaneous assignment of instances to one class or the other and clustering instances within each class, our method effectively captures non-witnesses within the positive bags from their clustering relationships to other instances. Figure 1 illustrates this idea.

We discover the latent positive bag instance labels by non-parametrically modeling the distributions of both classes, while simultaneously assigning the positive bag instances to the most appropriate class. To capture almost arbitrarily complex data distributions, we model the class distributions as mixture of a potentially very large (determined by data and the Dirichlet process prior) number of Gaussians with full covariance. The Dirichlet process prior on the mixture weights addresses the model selection problem, which is in our context the question of how many clusters to use.

We infer the class distribution parameters and positive bag instance labels by an efficient constrained variational inference procedure. For a fixed configuration of positive bag instance labels, we update class distribution parameters as in variational inference of standard Dirichlet process mixtures of Gaussians. Then keeping class distribution parameters fixed, we assign each positive bag instance to the class that maximizes the total variational lower bound of

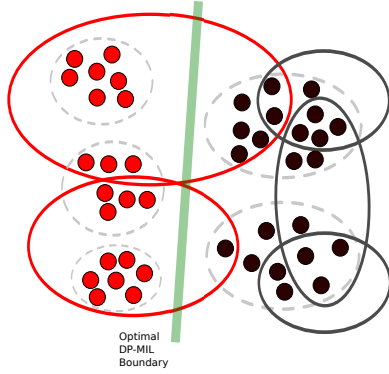


Figure 1: Dots, solid ellipses, and dashed ellipses indicate instances, bags, and clusters in a two dimensional feature space, respectively. Positive class is shown as red and negative class as black. DPMIL infers the label of a positive bag instance based on the class of the cluster that explains it best.

class distributions. This way, an increase in lower bound is guaranteed for all coordinate ascent updates, providing fast convergence.

We evaluate our method on 20 benchmark text categorization data sets, and on a novel application: finding Barrett’s cancer tumors in histopathology tissue images from bag labels. Our method improves the state-of-the-art in both of these applications in terms of instance-level prediction performance. Furthermore, differently from many existing MIL methods, the inferred data modes and cluster weights of our method enable enhanced interpretability. The source code of our method is publicly available ¹.

2 PRIOR ART

There exist several strategies for learning from weak supervision. One is *semi-supervised learning*, which suggests using large volumes of unlabeled data along with the limited labeled data to improve supervised learning performance [6]. *Active learning* is an alternative strategy that proposes learning from the smallest possible set of training samples selected by the model itself [24]. Another strategy is *self-taught learning* where abundant unlabeled data are available from a different but related task than the actual learning problem to be solved [20].

Multiple instance learning also aims to solve the weakly supervised learning problem by allowing supervision only for *groups of instances*. This learning setup has been first introduced by Dietterich et al. [7]. The authors propose detecting witnesses from the assumption that they lie in a

single axis parallel rectangle (APR) in the feature space.

MIL methods are built upon different heuristics. A group of methods iteratively choose one instance from each bag as a representative, and infer model parameters from this selected instance set. Based on the new model parameters, a new representative set is selected in the next iteration. Seminal examples of this approach are EMDD [30] and MI-SVM [2]. While the former learns a Gaussian density kernel on the representative instances, the latter trains a support vector machine (SVM) on them.

Another group of MIL methods calculate similarities between bag pairs by bag-level kernels, and train standard kernel learners, such as SVM, based on these bag similarities. MI Kernel [10] and mi-Graph [31] are seminal examples of this approach. The common property of these models is that they assume non-i.i.d. relationships between instances belonging to the same bag. There have been recent attempts to exploit within-bag correlations in more elaborate ways, such as Ellipsoidal MIL [15] and MIMN [11]. The former method represents each bag as an ellipsoid and learns a max-margin classifier that obeys the multiple instance constraints. The latter models the within-bag relationships by a Markov Random Field whose unary potentials are determined by the output of a linear instance-level classifier and clique (bag) potentials are calculated from the unary potentials subject to the multiple instance constraints. These methods are typically both effective and efficient. However, they are not applicable to instance level prediction due to the central non-i.i.d bag instances assumption.

MIL as semi-supervised learning. MIL can be formulated as a semi-supervised learning problem by assigning latent variables to positive bag instances and inferring them subject to the multiple instance constraints [8]. mi-SVM [2] applies this principle to the SVM formulation. GPMIL [14] and Bayesian Multiple Instance RVM [21] apply it to the Gaussian process classifier and the relevance vector machine, respectively, by adapting the likelihood function to MIL.

Generative MIL models. The semi-supervised learning approach has also been adopted by some generative methods that model the class distributions and infer the label of each positive bag instance based on which of these two distributions explain that instance with higher likelihood [1,8]. Foulds et al. [8] model each class distribution by a Gaussian density with isotropic or diagonal covariance, and learn the latent positive bag instances without employing the multiple instance constraints on the training data. Adel et al. [1], on the other hand, provide a generic framework that enforces the multiple instance constraint in the hard assignment of instances to classes. They model class distributions by a Gaussian density and Gaussian copula. We fol-

¹<http://hci.iwr.uni-heidelberg.de/Staff/mkandemi/>

low this line of research, and extend the existing work by i) using a richer family of distributions (potentially infinite mixtures of Gaussians with full covariance), while ii) keeping the multiple instance constraints and also providing an efficient variational inference procedure, and iii) making instance rather than bag level predictions.

Applications. Recent applications of MIL include diabetic retinopathy screening [19], visual saliency estimation [27] as well as content-based object detection and tracking [23]. MIL is also useful in drug activity prediction where each molecule constitutes a bag, each configuration of a molecule an instance, and binding of any of these configurations to the desired target is treated as a positive label, as first introduced by Dietterich et al. [7]. More recent applications of MIL to this problem include finding the interaction of proteins with Calmodulin molecules [18], and finding bioactive conformers [9]. Xu et al. [28, 29] apply MIL to tissue core (bag) level diagnosis of prostate cancer from histopathology images, where they combine multi-instance boosting [25] and clustering. There does not exist any prior work that focuses on locating tumors from tissue core level supervision, which we do in this paper as a case study.

Instance-level MIL prediction. There exist few studies focusing on instance prediction within the MIL setting. The first principled attempt towards this direction has been made by Zhou et al. [32]. The authors introduce a variant of Citation k-NN, called CkNN-ROI. This method chooses one instance from each positive bag as the *key instance* that determines the bag label based on how well it predicts the training bag labels by nearest neighbor matching, and ignores the other instances. Li et al. [16] detect key instances by a large margin method called KI-SVM. This method extends MI-SVM by binary latent variables assigned to each positive bag instance, which identify strictly one key instance per positive bag, and filter other instances out. The authors propose two variants of their method: i) Bag KI-SVM that has one slack variable per negative bag, and ii) Instance KI-SVM that has one slack variable per negative bag *instance*. Liu et al. [17] later propose detecting multiple key instances per positive bag by another variant of Citation kNN that learns a voting function from training bags. These models are shown to be effective in region-of-interest detection in natural scene images and text categorization. In this paper, we target the same learning problem, and empirically show that rich modeling of class distributions leads to better prediction performance.

3 THE MODEL

Let \mathbf{X} be a data set consisting of B bags $\mathbf{X} = [\mathbf{X}_1, \dots, \mathbf{X}_B]$ indexed by b , and $\mathbf{y} = [y_1, \dots, y_B]$ be the vector of the corresponding binary bag labels $y_b \in$

$\{-1, +1\}$. Each bag $\mathbf{X}_b = [\mathbf{x}_{b1}, \dots, \mathbf{x}_{bN_b}]$ consists of N_b instances. We assume that each instance is associated with a binary latent variable $r_{bn} \in \{-1, +1\}$ representing the label of the instance. We further assume that the positive instances in the data set ($r_{bn} = +1$) come from distribution $p(\mathbf{x}_{bn}|\boldsymbol{\theta}_{+1})$, and the negative instances ($r_{bn} = -1$) come from distribution $p(\mathbf{x}_{bn}|\boldsymbol{\theta}_{-1})$, parameterized by $\boldsymbol{\theta}_{+1}$ and $\boldsymbol{\theta}_{-1}$, respectively. Both of these two distributions are Gaussian mixture models with full covariance and with Dirichlet process priors on mixture weights. The generative process of our model is

$$\begin{aligned} p(\mathbf{v}_l) &= \prod_{k=1}^K \text{Beta}(v_{lk}|1, \alpha), & \forall l \\ p(z_{lbn}|\mathbf{v}_l) &= \text{Mult}(z_{lbn}|\pi_{l1}, \dots, \pi_{lK}), & \forall l, b, n \\ p(\boldsymbol{\Lambda}_{lk}) &= \mathcal{W}(\boldsymbol{\Lambda}_{lk}|\mathbf{W}_0, \nu_0), & \forall l, k \\ p(\boldsymbol{\mu}_{lk}|\boldsymbol{\Lambda}_{lk}) &= \mathcal{N}(\boldsymbol{\mu}_{lk}|\mathbf{m}_0, (\beta_0 \boldsymbol{\Lambda}_{lk})^{-1}), & \forall l, k, \\ p(\mathbf{x}_{bn}|\boldsymbol{\mu}, \boldsymbol{\Lambda}, z_{lbn}, r_{bn}) &= \\ & \prod_{l \in \{-1, +1\}} \prod_{k=1}^K \mathcal{N}(\mathbf{x}_{bn}|\boldsymbol{\mu}_{lk}, \boldsymbol{\Lambda}_{lk}^{-1})^{\mathbf{1}(z_{lbn}=k) \cdot \mathbf{1}(r_{bn}=l)}, & \forall b, n, \\ p(y_b = +1|\mathbf{r}) &= 1 - \prod_{n=1}^{N_b} (1 - \mathbf{1}(r_{bn} = +1)), & \forall b \end{aligned}$$

where the hyperparameters of the model are $\{\nu_0, \mathbf{W}_0, \mathbf{m}_0, \beta_0, \alpha\}$. The function $\mathbf{1}(\cdot)$ is the indicator function which returns 1 if its argument is true, and 0 otherwise. $\text{Mult}(\cdot|\cdot)$, $\text{Beta}(\cdot|\cdot)$, $\mathcal{N}(\cdot|\cdot)$ and $\mathcal{W}(\cdot|\cdot)$ denote the multinomial mass function, and Beta, Gaussian and Wishart distribution densities, respectively. K is the number of clusters, and k is the related index; $l \in \{-1, +1\}$ indexes the two class densities; $\pi_{lk} = v_{lk} \prod_{j=1}^{k-1} (1 - v_{lj})$ is the stick breaking prior over cluster assignments z_{lbn} . The vector \mathbf{Z}_l contains cluster-assignment weights z_{lbn} . The sets $\boldsymbol{\mu} = \{\boldsymbol{\mu}_{-11}, \dots, \boldsymbol{\mu}_{-1K}, \boldsymbol{\mu}_{+11}, \dots, \boldsymbol{\mu}_{+1K}\}$ and $\boldsymbol{\Lambda} = \{\boldsymbol{\Lambda}_{-11}, \dots, \boldsymbol{\Lambda}_{-1K}, \boldsymbol{\Lambda}_{+11}, \dots, \boldsymbol{\Lambda}_{+1K}\}$ contain the mean and inverse covariance of all clusters in the model, respectively. The vector \mathbf{r} has class-assignment variables for all instances in its entries, and $\mathbf{r}_{-r_{bn}}$ has the same for all instances except r_{bn} . The set \mathbf{r}_b has the class-assignment variables of bag b . If $y_b = -1$ is observed, it is also observed that $r_{bn} = -1$ for all instances of bag b . If $y_b = +1$ is observed, r_{bn} for bag instances of b are latent, hence are inferred from data. We refer to this model as *Dirichlet process multiple instance learning (DPMIL)*. Figure 2 illustrates the model in plate notation.

3.1 Inference

Following the probabilistic paradigm, for inference of the model above, we aim to maximize the marginal likelihood $p(\mathbf{X}, \mathbf{y}|\mathbf{z})$ with respect to the class assignments \mathbf{z} subject

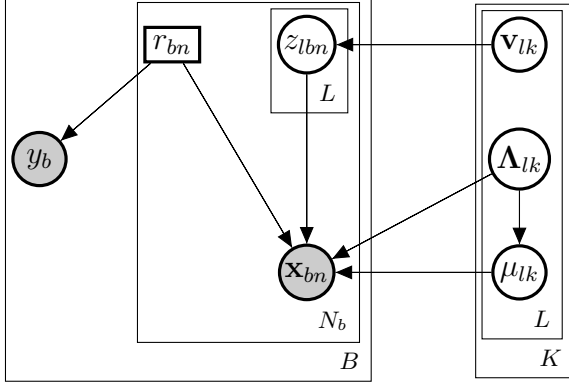


Figure 2: The generative process of DPMIL in plate notation. Shaded nodes denote observed, and unshaded nodes denote latent variables that are inferred by constrained variational Bayes. Note that r_{bn} is a discrete binary latent variable without a prior. Hence it is denoted by a rectangle.

to the multiple instance constraints

$$\begin{aligned} & \underset{\mathbf{r}}{\text{maximize}} \quad p(\mathbf{X}, \mathbf{y} | \mathbf{r}) \\ & \text{s.t.} \quad \max(\mathbf{r}_b) = y_b, \quad \forall b. \end{aligned} \quad (1)$$

Let \mathbf{r}_* be a solution to the optimization problem (1), we can define the divergence from the optimal configuration \mathbf{r}_* as

$$D(\mathbf{r}) = \log p(\mathbf{X}, \mathbf{y} | \mathbf{r}_*) - \log p(\mathbf{X}, \mathbf{y} | \mathbf{r}).$$

It is easy to see that $D(\mathbf{r}) \geq 0$ for any \mathbf{r} and $D(\mathbf{r}) = 0$ if $\mathbf{r} = \mathbf{r}_*$.

For a given configuration \mathbf{r} , calculating $p(\mathbf{X}, \mathbf{y} | \mathbf{r})$ is intractable. Hence, we approximate the posterior p a factorized distribution q

$$\begin{aligned} & p(\mathbf{Z}, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \mathbf{v}_{-1}, \mathbf{v}_{+1} | \mathbf{X}, \mathbf{r}) \\ &= \left(\prod_{l \in \{-1, +1\}} \prod_{b=1}^B \prod_{n=1}^{N_b} q(z_{lbn} | \mathbf{r}) \right) \\ & \times \left(\prod_{l \in \{-1, +1\}} \prod_{k=1}^K q(\boldsymbol{\mu}_{lk}, \boldsymbol{\Lambda}_{lk} | \mathbf{r}) q(\mathbf{v}_{lk} | \mathbf{r}) \right). \end{aligned}$$

Let $\boldsymbol{\theta} = \boldsymbol{\theta}_{-1} \cup \boldsymbol{\theta}_{+1}$ denote the set of all parameters and latent variables of both class distributions. Following the standard variational Bayes formulation we can decompose $p(\mathbf{X}, \mathbf{y} | \mathbf{r})$ as

$$\log p(\mathbf{X}, \mathbf{y} | \mathbf{r}) = \mathcal{L}(\boldsymbol{\theta} | \mathbf{r}) + KL(q || p)$$

where

$$\mathcal{L}(\boldsymbol{\theta} | \mathbf{r}) = \mathbb{E}_q[\log p(\mathbf{X}, \mathbf{y}, \boldsymbol{\theta} | \mathbf{r})] - \mathbb{E}_q[\log q(\boldsymbol{\theta} | \mathbf{r})]$$

is the variational lower bound and $KL(\cdot || \cdot)$ is the Kullback-Leibler divergence between the true posterior p and the approximate posterior q . Similarly to above, $KL(q || p) \geq 0$

for all q and $KL(q || p) = 0$ if and only if $q = p$. Combining these two facts, we have

$$\log p(\mathbf{X}, \mathbf{y} | \mathbf{r}_*) = \mathcal{L}(\boldsymbol{\theta} | \mathbf{r}) + \underbrace{KL(q || p) + D(\mathbf{r})}_{E(q, \mathbf{r})}$$

where the divergence term $E(q, \mathbf{r})$ approaches 0 as q and \mathbf{r} approach optimal values. Hence, we can perform inference by

$$\begin{aligned} & \underset{\mathbf{r}, \boldsymbol{\theta}}{\text{maximize}} \quad \mathcal{L}(\boldsymbol{\theta} | \mathbf{r}) \\ & \text{s.t.} \quad \max(\mathbf{r}_b) = y_b, \quad \forall b. \end{aligned}$$

which has the same global optimum as the optimization problem (1). This problem can be solved by coordinate ascent. Keeping \mathbf{r} fixed, model parameters $\boldsymbol{\theta}$ can be updated as in standard variational Bayes. Let $\boldsymbol{\psi}_j \subset \boldsymbol{\theta}$ be a subset of model parameters corresponding to a factor of q , the best possible update for this factor can be calculated by

$$\frac{\partial \mathcal{L}}{\partial q(\boldsymbol{\psi}_j)} = \mathbb{E}_{q(\boldsymbol{\theta}_{-\boldsymbol{\psi}_j})}[\log p(\mathbf{X}, \mathbf{y}, \boldsymbol{\theta} | \mathbf{r})] - \log q(\boldsymbol{\psi}_j) - 1 = 0.$$

Hence, the update rule becomes

$$q(\boldsymbol{\psi}_j) = \exp \left\{ \mathbb{E}_{q(\boldsymbol{\theta}_{-\boldsymbol{\psi}_j})}[\log p(\mathbf{X}, \mathbf{y}, \boldsymbol{\theta} | \mathbf{r})] \right\}. \quad (2)$$

Consequently, keeping $\boldsymbol{\theta}$ fixed, \mathbf{r} can be updated by

$$r_{bn}^{(t+1)} = \underset{l \in \{-1, +1\}}{\text{argmax}} \mathcal{L}(\boldsymbol{\theta} | \mathbf{r}_{-bn}^{(t)}, r_{bn} = l). \quad (3)$$

The cases that violate the multiple instance constraint $\max(\mathbf{r}_b^{(t+1)}) = y_b$ can be resolved by flipping one of the instances of bag b that had a positive label at iteration (t) back to positive. The fact that Equations (2) and (3) both increase \mathcal{L} and that $E(q, \mathbf{r}) \geq 0$ bring out fast convergence to a local maximum in practice, as experimented in Section 4.3. The overall inference procedure is given in Algorithm 1, and the detailed update equations are available in Appendix 1.

3.2 Prediction

For a new bag $\mathbf{X}_b^* = [\mathbf{x}_{b1}^*, \dots, \mathbf{x}_{bN_b}^*]$, instance-level prediction can be done by

$$\hat{r}_{bn} \leftarrow \underset{l \in \{-1, +1\}}{\text{argmax}} p(\mathbf{x}_{bn}^* | \mathbf{X}, \mathbf{y}, \mathbf{r}, y_{bn}^* = l),$$

where

$$p(\mathbf{x}_{bn}^* | \mathbf{X}, \mathbf{y}, \mathbf{r}, y_{bn}^* = l) = \int q(\boldsymbol{\theta}_l | \mathbf{X}, \mathbf{y}, \mathbf{r}) p(\mathbf{x}_{bn}^* | \boldsymbol{\theta}_l) d\boldsymbol{\theta}_l,$$

which corresponds to the standard predictive density for DP Gaussian mixtures as given in [4]. The extended formula of the predictive density for fixed \mathbf{r} is given in Appendix 1.

Algorithm 1 Constrained variational inference for DPMIL

Input: Data $\mathbf{X} = [\mathbf{X}_1, \dots, \mathbf{X}_B]$,
Bag labels $\mathbf{y} = \{y_1, \dots, y_b\}$
repeat
 $\backslash\backslash$ Initialize instance class labels
 $r_{bn} = y_b, \quad \forall b, n$
 $\backslash\backslash$ Update the class distributions given the current \mathbf{r}
 for $\psi_j \in \theta$ **do**
 $q(\psi_j|\mathbf{r}) \leftarrow \exp \left\{ (\mathbb{E}_{q(\theta-\psi_j)} [\log p(\mathbf{X}, \mathbf{y}, \theta|\mathbf{r})]) \right\}$
 end for
 $\backslash\backslash$ Update \mathbf{r} given the class distributions
 for $b \in \{j|y_j = +1\}$ **do**
 for $n = 1$ **to** N_B **do**
 $r_{bn}^{(t+1)} \leftarrow \underset{l \in \{-1, +1\}}{\operatorname{argmax}} \mathcal{L}(\theta|\mathbf{r}_{-r_{bn}}^{(t)}, r_{bn} = l)$
 end for
 $\backslash\backslash$ Resolve constraint violation
 if $\max(\mathbf{r}_b) = -1$ **then**
 $r_{bj}^{(t+1)} \leftarrow +1$, for any $j \in \{r_{bj}^{(t)} = +1\}$
 end if
 end for
until *convergence*

3.3 Relationship to existing models

DPMIL has the following connections to some of the existing methods:

- **mi-SVM** [2]: DPMIL and mi-SVM can be viewed as generative-discriminative pairs [12]. The two models find similar labels for positive bag instances when classes are separable. DPMIL additionally finds the clusters of both positive and negative instances.
- **EMDD** [30]: EMDD learns a class-conditional distribution $p(y_b = +1|\mathbf{X}_b)$ in a *discriminative* manner by applying a *single* Gaussian kernel on the most representative *subset* of training instances. DPMIL explains the *generative* process of *all* training instances by *multiple* Gaussian densities.
- **QDA**: Our method extends Quadratic Discriminant Analysis (QDA) in three aspects: i) DPMIL fits multiple Gaussians on each class distribution, while QDA fits only one. ii) DPMIL employs priors over mean and covariance, while QDA performs maximum likelihood estimation, following the frequentist paradigm. iii) DPMIL explains bag labels keeping the multiple instance constraints, while QDA performs single-instance learning.
- **MIMM** [8]: This model is a special case of DPMIL. In particular, when $K = 1$, uninformative priors are used for mixture coefficients \mathbf{Z} and multiple instance constraints are ignored, DPMIL reduces to MIMM.

Quadratic Discriminant Analysis (QDA) is the single-instance version of MIMM.

4 RESULTS

We evaluate the instance prediction performance of our method on two applications: i) web page categorization, and ii) Barrett’s cancer diagnosis. For both experiments, we set cluster count K to 20 (per class), ν_0 to $D + 1$, where D is the dimensionality of the data, \mathbf{W}_0 to the inverse empirical covariance of the data, \mathbf{m}_0 to the empirical mean of the data, β_0 to 1, and the concentration parameter α to 2, which is chosen as the smallest integer larger than the uninformative case ($\alpha = 1$). This value is not manually tuned. Other choices of α are observed not to affect the outcome significantly. We set maximum iteration count to 100.

We compare DPMIL to three MIL and two key instance detection algorithms: mi-SVM [2], MI-SVM [2], GPMIL [14], Bag KI-SVM [16], and Instance KI-SVM [16]. Models such as mi-Graph [31], iAPR [7], EMDD [30], Citation k-NN [26], MILBoost [25], and MIMM [8] are observed to perform worse than the list above, hence are not reported in detail. For all kernelizable models, the radial basis function (RBF) kernel is used. Hyperparameters of the competing models are learned by cross-validation.

4.1 20 text categorization data sets

As a benchmarking study, we evaluate DPMIL on the public *20 Newsgroups* database that consists of 20 text categorization data sets. Each data set consists of 50 positive and 50 negative bags. Positive bags have on average 3 % of their instances from the target category, and the rest from other categories. Each instance in a bag is the top 200 TF-IDF representation of a post. We reduce the dimensionality to 100 by Kernel Principal Component Analysis (KPCA) with an RBF kernel with a length scale of $\sqrt{100}$, following the heuristic of Chang et al [5]. We evaluate the generalization performance using 10-fold cross validation with the standard data splits. We use Area Under Precision-Recall Curve (AUC-PR) as the performance measure due to its insensitivity to class imbalance. Table 1 lists the performance scores of models in comparison for the 20 data sets. We report average AUC-PR of two comparatively recent methods, VF and VF_r, on the same database from [17] Table 5², for which public source code is not available. Our method gives the highest instance prediction performance in 18 of the 20 data sets, and its average performance throughout the database is 3 percentage points higher than the state-of-the-art VF method.

Table 1: Area Under Precision-Recall Curve (AUC-PR) scores of methods on the 20 *Newsgroups* database for instance prediction. DPMIL outperforms the other MIL models in 18 out of 20 data sets. B-KI-SVM and I-KI-SVM stand for Bag KI-SVM and Instance KI-SVM, respectively.

Data set	DPMIL	VF	VFr	B-KISVM	miSVM	I-KISVM	GPML	MISVM
alt.atheism	0.67	-	-	0.68	0.53	0.46	0.44	0.38
comp.graphics	0.79	-	-	0.47	0.65	0.62	0.49	0.07
comp.os.ms-windows.misc	0.51	-	-	0.38	0.42	0.14	0.36	0.03
comp.sys.ibm.pc.hardware	0.67	-	-	0.31	0.57	0.38	0.35	0.10
comp.sys.mac.hardware	0.76	-	-	0.39	0.56	0.64	0.54	0.27
comp.windows.x	0.73	-	-	0.37	0.56	0.35	0.36	0.04
misc.forsale	0.45	-	-	0.29	0.31	0.25	0.33	0.10
rec.autos	0.76	-	-	0.45	0.51	0.42	0.38	0.34
rec.motorcycles	0.69	-	-	0.52	0.09	0.61	0.46	0.27
rec.sport.baseball	0.74	-	-	0.52	0.18	0.41	0.38	0.22
rec.sport.hockey	0.91	-	-	0.66	0.27	0.64	0.43	0.75
sci.crypt	0.68	-	-	0.47	0.57	0.26	0.31	0.32
sci.electronics	0.90	-	-	0.42	0.83	0.65	0.71	0.34
sci.med	0.73	-	-	0.55	0.37	0.44	0.32	0.44
sci.space	0.70	-	-	0.51	0.46	0.33	0.32	0.20
soc.religion.christian	0.72	-	-	0.53	0.05	0.45	0.45	0.40
talk.politics.guns	0.64	-	-	0.43	0.57	0.32	0.38	0.01
talk.politics.mideast	0.80	-	-	0.60	0.77	0.49	0.46	0.60
talk.politics.misc	0.60	-	-	0.50	0.61	0.38	0.29	0.30
talk.religion.misc	0.51	-	-	0.32	0.08	0.34	0.32	0.04
Average	0.70	0.67	0.59	0.47	0.45	0.43	0.40	0.26

Table 2: Barrett’s cancer diagnosis accuracy and F1 score of models in comparison. DPMIL outperforms the second best model by 6 percentage points in accuracy and 3 percentage points in F1 score. Instance level supervision performance is provided in the bottom row for reference.

Method	Accuracy (%)	F1 Score
DPMIL	71.8	0.74
GPML	65.8	0.54
I-KISVM	65.4	0.45
B-KISVM	64.7	0.48
mi-SVM	62.7	0.71
MISVM	46.9	0.64
SVM	83.5	0.82

4.2 Barrett’s cancer diagnosis

Biopsy imaging is a widely used cancer diagnosis technique in clinical pathology [22]. A sample is taken from the suspicious tissue, stained with hematoxylin & eosin, which dyes nuclei, stroma, lumen, and cytoplasm to different colours. Afterwards, the tissue is photographed under a microscope, and a pathologist examines the resultant image for diagnosis. In many cases, diagnosis of one patient requires careful scanning of several tissue slides of extensive

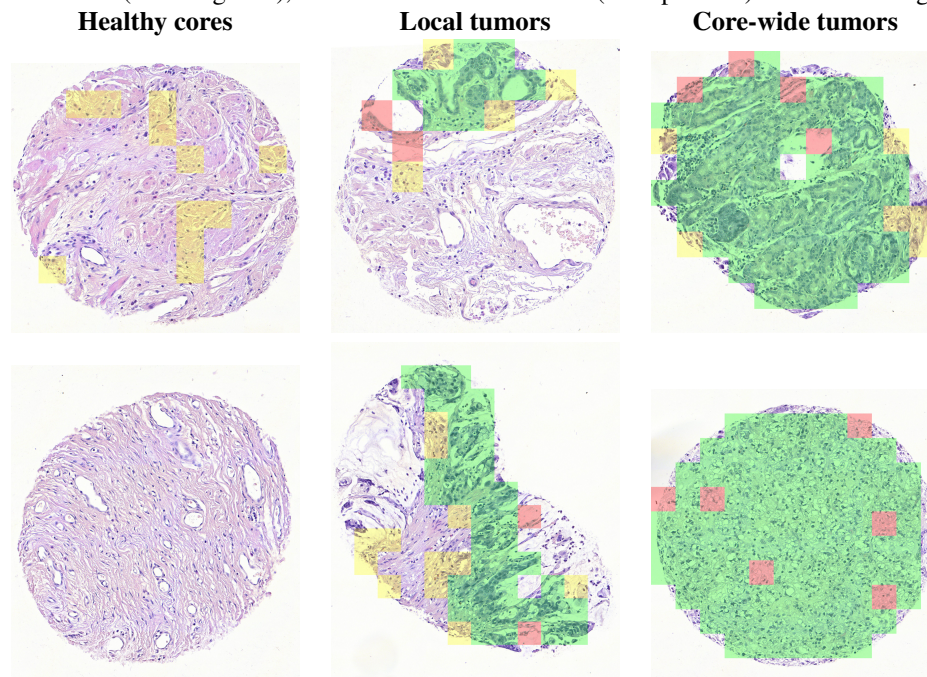
sizes. Considerable time could be saved by an algorithm that finds the tumors and leads the pathologist to tumorous regions.

We evaluate DPMIL in the task of finding Barrett’s cancer tumors in human esophagus tissue images from image-level supervision. Our data consists of 210 tissue core images (143 cancer and 67 healthy) taken from 97 Barrett’s cancer patients. We treat tumor regions drawn by expert pathologists as ground truth. We split each tissue core (with average size of 2179x1970 pixels) into a grid of 200x200 pixel patches. We represent each patch by a 738-dimensional feature vector of SIFT descriptors, local binary patterns with 20×20 -pixel cells, intensity histogram of 26 bins for each of the RGB channels, and the mean of the features described in [13] for cells lying in that patch. The data set includes 14303 instances, 53.4% of which are cancerous. We treat each image as a bag and each patch belonging to that image as an instance. A bag is labeled as positive if it includes tumor, and negative otherwise. Similarly to above, we reduce the data dimensionality to 30 by KPCA with an RBF kernel having a length scale of $\sqrt{30}$. We evaluate generalization performance by 4-fold cross-validation over bags. We repeated this procedure 5 times.

The patch-level diagnosis performance comparison of models is given in Table 2. Prediction performance of DPMIL lies in the middle of the chance level of 53.4% and the upper bound of 83.5% which is reached by patch-level

² Liu et al. [17] report 0.42 AUC-PR for KI-SVM and 0.41 AUC-PR for mi-SVM in Table 5.

Figure 3: Patch prediction results on sample tissue core images. **Green:** correctly detected cancer (true positive), **Red:** Missed detection of cancer (false negative), **Yellow:** False cancer alarm (false positive). **Rest:** True negative.



training of an SVM with RBF kernel. DPMIL clearly outperforms existing models both in prediction accuracy and F1 score (harmonic mean of precision and recall). Figure 3 shows prediction results of DPMIL on six sample tissue cores (bags) with different proportions of tumor. DPMIL produces few false positives for the healthy tissues (left-most column), detects local tumors with reasonable accuracy (middle columns), and produces few false negatives for tissue cores covered entirely by tumor (right-most column).

Figure 4 shows the mixture weights of the clusters for the class distributions averaged over data splits. The *healthy* class is dominated by a single cluster due to the relatively uniform structure of a healthy esophagus tissue. On the other hand, for the *cancer* class, the weights are more evenly distributed among five clusters. This result is consistent with the fact that the data set includes images from various grades of cancer. Each grade of cancer causes a different visual pattern in the tissue, resulting in a multi-modal distribution of tumor patches. As shown in Figure 5, clusters capture meaningful visual structures. Patches in the first row correspond to a stage of Barrett’s cancer where cells form circular structures called *glands* which do not exist in a healthy esophagus tissue. The second row illustrates samples of cells with faded color, and in the third row the tissue is covered by an overly high population of poorly differentiated cells.

4.3 Learning rate and computational time

Weak supervision often emerges as a necessity for analyzing big data. Hence, computational efficiency of an MIL model is of key importance for feasibility for real-world scenarios. To this end, we provide an empirical analysis of the learning rate and the training time of DPMIL. As shown in Figure 6, the variational lower bound $\log \mathcal{L}(\theta|\mathbf{r})$ exhibits a sharp increase in the first few iterations, and saturates within 50 iterations.

Figure 6: Evolution of the variational lower bound $\log \mathcal{L}(\theta|\mathbf{r})$ throughout training iterations for the Barrett’s cancer data set. DPMIL exhibits a steep learning curve and converges in less than 50 iterations.

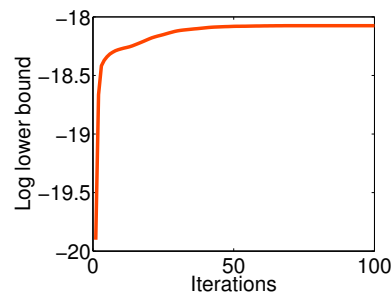


Table 3 shows the average training times of the models in comparison for one data split. Thanks to its Bayesian nonparametric nature, DPMIL does not require a cross-validation stage for model selection, unlike the other mod-

Figure 4: Cluster mixture coefficients for *cancer* ($y_b = +1$) and *healthy* ($y_b = -1$) in the Barrett’s cancer data set. The healthy class distribution is dominated by a single mode unlike the cancer class distribution, supporting that a healthy tissue has a more even look than the cancer class which includes images belonging to various levels of cancer.

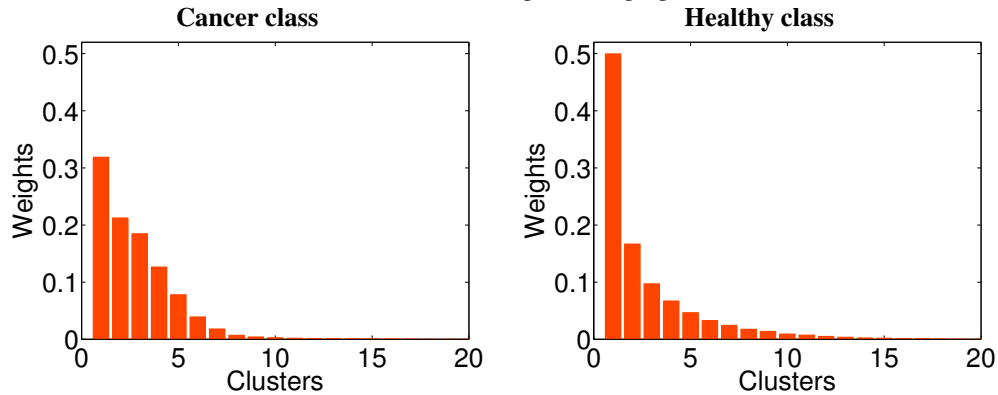
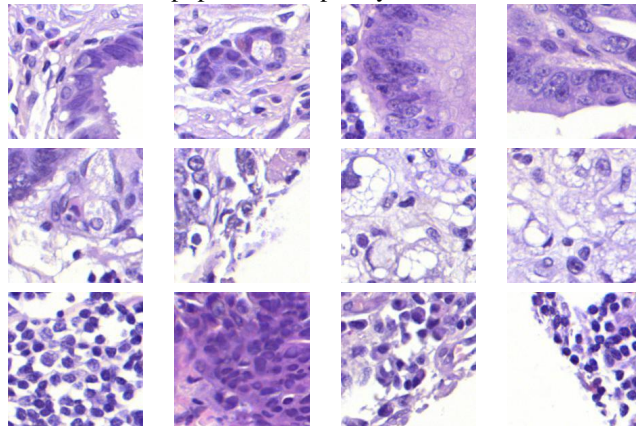


Figure 5: Sample patches from three different clusters (one in each row) of the *cancer* class. Each patch belongs to a different image. First cluster shows glandular formations of cancer cells, second cluster contains single cancer cells with faded color, and third cluster shows increased population of poorly differentiated cancer cells.



els. To avoid variability due to the desired level of detail in hyperparameter tuning (grid resolution and number of validation splits) which could lead to unfair comparison, we excluded the cross-validation time for the competing models. As a result of its steep learning rate, DPMIL provides reasonable training time, ranking as the most efficient model in text categorization and third in Barrett’s cancer diagnosis.

5 DISCUSSION

Multiple instance learning methods have long been developed and evaluated for bag label prediction. In this paper, we focus on the harder problem of instance level prediction from bag level training. We approach the problem from a semi-supervised learning perspective, and attempt to discover the unknown labels of positive bag instances by rich modeling of class distributions in a generative manner. We model these distributions by Gaussian mixture models with full covariance to handle complex multimodal cases. To

Table 3: Training times (in seconds) of models in comparison for one data split. Thanks to the efficient variational inference procedure, DPMIL can be trained in reasonable time.

Model name	Text categorization	Barrett’s cancer
DPMIL	2.9	44.7
KISVM-B	11.0	107.7
mi-SVM	12.2	126.6
KISVM-I	10.1	15.3
GPMIL	90.5	1491.7
MISVM	4.1	10.8

avoid the model selection problem (i.e. predetermination of the number of data modes), we apply Dirichlet process priors over mixture coefficients.

As experimented in a large set of benchmark data sets and one cancer diagnosis application, our method clearly improves the state-of-the-art in instance classification from

weak labels. We attribute this improvement to the effectiveness of the *let the data speak* attitude in semi-supervised learning: The model discovers the unknown positive bag instance labels by assigning them to the class that explains the data generation process better (i.e. the class that increases the variational lower bound more). Of the other methods in our comparison, mi-SVM, VF, and KISVM are ignorant about the class distributions. The remaining methods are tailored for predicting bag, but not instance labels.

Generative modeling of data is commonly undesirable in standard pattern classification tasks, as a result of Vapnik's razor principle³. However, our results imply that generative data distribution modeling turns out to be an effective strategy when weak supervision is an additional source of uncertainty.

Modeling class distributions with mixture models brings enhanced interpretability as a by-product. Analysis of inferred clusters may provide additional information, or may support further modeling decisions. Even though we restrict our analysis to binary classification for illustrative purposes, extension of our method to multiclass cases is simply a matter of increasing the number of Gaussian mixture models from two to a desired number of classes.

Appendix 1: Variational update equations and predictive density

Variational update equations of the approximate posterior q correspond to those of the Gaussian mixture model as described in [3] where the Dirichlet prior on mixture weights are replaced by a Dirichlet process prior and instances are assigned to the appropriate distribution by indicator functions $\mathbf{1}(\cdot)$.

For $q(\nu_{lk}) = \text{Beta}(\gamma_{lk}^1, \gamma_{lk}^2)$,

$$\begin{aligned}\gamma_{lk}^1 &= 1 + \sum_{b=1}^B \sum_{n=1}^{N_b} q(z_{lbn} = k) \mathbf{1}(r_{bn} = l), \\ \gamma_{lk}^2 &= \alpha + \sum_{b=1}^B \sum_{n=1}^{N_b} q(z_{lbn} > k) \mathbf{1}(r_{bn} = l).\end{aligned}$$

For $q(z_{lbn} = k) = \text{Mult}(\tau_{lbn}^1, \dots, \tau_{lbn}^K)$,

$$\begin{aligned}\tau_{lbn}^k &\leftarrow \left(\Psi(\gamma_{lk}^1) - \Psi(\gamma_{lk}^1 + \gamma_{lk}^2) + \sum_{j=1}^{k-1} (\Psi(\gamma_{lj}^2) - \Psi(\gamma_{lj}^1 + \gamma_{lj}^2)) \right. \\ &\quad \left. + \sum_{i=1}^D \Psi\left(\frac{\nu_{lk} + 1 - i}{2}\right) + D \log(2) + \log |\mathbf{W}_{lk}| - \frac{D}{2} \log(2\pi) \right. \\ &\quad \left. - \frac{D}{2} \beta_{lk}^{-1} - \frac{1}{2} \nu_{lk} (\mathbf{x}_{bn} - \mathbf{m}_{lk})^T \mathbf{W}_{lk} (\mathbf{x}_{bn} - \mathbf{m}_{lk}) \right) \mathbf{1}(r_{bn} = l).\end{aligned}$$

³**Vapnik's razor principle:** When solving a (learning) problem of interest, do not solve a more complex problem as an intermediate step.

For $q(\boldsymbol{\mu}_{lk}, \boldsymbol{\Lambda}_{lk}) = \mathcal{N}(\boldsymbol{\mu}_{lk} | \mathbf{m}_{lk}, (\beta_{lk} \boldsymbol{\Lambda}_{lk}^{-1})) \mathcal{W}(\boldsymbol{\Lambda}_{lk} | \mathbf{W}_{lk}, \nu_{lk})$, where

$$\begin{aligned}\beta_{lk} &= \beta_0 + N_{lk}, \\ \mathbf{m}_{lk} &= \beta_{lk}^{-1} (\beta_0 \mathbf{m}_0 + N_{lk} \bar{\mathbf{x}}_{lk}), \\ \mathbf{W}_{lk}^{-1} &= \mathbf{W}_0^{-1} + N_{lk} \mathbf{S}_{lk} + \frac{\beta_0 N_{lk}}{\beta_0 + N_{lk}} (\bar{\mathbf{x}}_{lk} - \mathbf{m}_0)(\bar{\mathbf{x}}_{lk} - \mathbf{m}_0)^T, \\ \nu_{lk} &= \nu_0 + N_{lk} + 1.\end{aligned}$$

Here,

$$\begin{aligned}N_{lk} &= \sum_b^B \sum_{n=1}^{N_b} \mathbf{1}(r_{bn} = l) q(z_{lbn} = k), \\ \bar{\mathbf{x}}_{lk} &= \frac{1}{N_{lk}} \sum_b^B \sum_{n=1}^{N_b} \mathbf{1}(r_{bn} = l) q(z_{lbn} = k) \mathbf{x}_{bn}, \\ \mathbf{S}_{lk} &= \frac{1}{N_{lk}} \sum_b^B \sum_{n=1}^{N_b} \mathbf{1}(r_{bn} = l) q(z_{lbn} = k) (\mathbf{x}_{bn} - \bar{\mathbf{x}}_{lk})(\mathbf{x}_{bn} - \bar{\mathbf{x}}_{lk})^T.\end{aligned}$$

For an inferred configuration $\hat{\mathbf{r}}$, the predictive density of DPMIL is identical to that of a standard Gaussian mixture model as given in [3]

$$\begin{aligned}p(\mathbf{x}_{bn}^* | \mathbf{X}, \mathbf{y}, \hat{\mathbf{r}}, y_{bn}^* = l) &= \int q(\boldsymbol{\theta}_l | \mathbf{X}, \mathbf{y}, \hat{\mathbf{r}}) p(\mathbf{x}_{bn}^* | \boldsymbol{\theta}_l) d\boldsymbol{\theta}_l, \\ &= \frac{1}{\hat{\pi}_l} \sum_{k=1}^K \pi_{lk} St\left(\mathbf{x}_{bn}^* \middle| \mathbf{m}_k, \frac{(\nu_k + 1 - D)\beta_k}{1 + \beta_k} \mathbf{W}_k, \nu_k + 1 - D\right),\end{aligned}$$

where $\hat{\pi}_{lk} = \sum_{k=1}^K \pi_{lk}$ and $St(\cdot | \cdot, \cdot, \cdot)$ is the Student's t density function.

References

- [1] T. Adel, B. Smith, R. Urner, D. Stashuk, and D.J. Lizotte. Generative multiple-instance learning models for quantitative electromyography. In *UAI*, 2013.
- [2] S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In *NIPS*, 2003.
- [3] C.M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.
- [4] D.M. Blei and M.I. Jordan. Variational inference for Dirichlet process mixtures. *Bayesian Analysis*, 1(1):121–143, 2006.
- [5] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- [6] O. Chapelle, B. Schölkopf, A. Zien, et al. *Semi-supervised learning*. MIT Press, Cambridge, 2006.

- [7] T.G. Dietterich, R.H. Lathrop, and T. Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1):31–71, 1997.
- [8] J.R. Foulds and P. Smyth. Multi-instance mixture models and semi-supervised learning. In *SIAM Int’l Conf. Data Mining*, 2011.
- [9] G. Fu, X. Nan, H. Liu, R. Patel, P. Daga, Y. Chen, D. Wilkins, and R. Doerksen. Implementation of multiple-instance learning in drug activity prediction. *BMC Bioinformatics*, 13(Suppl 15):S3, 2012.
- [10] T. Gärtner, P.A. Flach, A. Kowalczyk, and A.J. Smola. Multi-instance kernels. In *ICML*, 2002.
- [11] H. Hajimirsadeghi, J. Li, G. Mori, M. Zaki, and T. Sayed. Multiple instance learning by discriminative training of markov networks. In *UAI*, 2013.
- [12] A. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *NIPS*, 2002.
- [13] M. Kandemir, A. Feuchtinger, A. Walch, and F.A. Hamprecht. Digital Pathology: Multiple instance learning can detect Barrett’s cancer. In *ISBI*, 2014.
- [14] M. Kim and F. De La Torre. Gaussian process multiple instance learning. In *ICML*, 2010.
- [15] G. Krummenacher, C.S. Ong, and J. Buhmann. Ellipsoidal multiple instance learning. In *ICML*, 2013.
- [16] Y.-F. Li, J.T. Kwok, I.W. Tsang, and Z.-H. Zhou. A convex method for locating regions of interest with multi-instance learning. In *Machine learning and knowledge discovery in databases*, pages 15–30. Springer, 2009.
- [17] G. Liu, J. Wu, and Z.-H. Zhou. Key instance detection in multi-instance learning. *Journal of Machine Learning Research-Proceedings Track*, 25:253–268, 2012.
- [18] F.A. Minhas and A. Ben-Hur. Multiple instance learning of Calmodulin binding sites. *Bioinformatics*, 28(18):i416–i422, 2012.
- [19] G. Quellec, M. Lamard, M.D. Abràmoff, E. Decencière, B. Lay, A. Erginay, B. Cochener, and G. Cazuguel. A multiple-instance learning framework for diabetic retinopathy screening. *Medical Image Analysis*, 2012.
- [20] R. Raina, A. Battle, H. Lee, B. Packer, and A.Y. Ng. Self-taught learning: transfer learning from unlabeled data. In *ICML*, 2007.
- [21] V.C. Raykar, B. Krishnapuram, J. Bi, M. Dundar, and R.B. Rao. Bayesian multiple instance learning: automatic feature selection and inductive transfer. In *ICML*, 2008.
- [22] R. Rubin and D.S. Strayer. *Rubin’s pathology: clinicopathologic foundations of medicine*. Lippincott Williams & Wilkins, 2008.
- [23] P. Sharma, C. Huang, and R. Nevatia. Unsupervised incremental learning for improved object detection in a video. In *CVPR*, 2012.
- [24] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. In *Journal of Machine Learning Research*, 2000.
- [25] P. Viola, J. Platt, and C. Zhang. Multiple instance boosting for object detection. *NIPS*, 2006.
- [26] J. Wang and J.D. Zucker. Solving multiple-instance problem: A lazy learning approach. *ICML*, 2000.
- [27] Q. Wang, Y. Yuan, P. Yan, and X. Li. Saliency detection by multiple-instance learning. *IEEE Trans. on Systems, Man and Cybernetics B*, 43(2):660 – 672, 2013.
- [28] Y. Xu, J. Zhang, E.-C. Chang, M. Lai, and Z. Tu. Context-constrained multiple instance learning for histopathology image segmentation. *Lecture Notes in Computer Science*, 7512:623–630, 2012.
- [29] Y. Xu, J.Y. Zhu, E. Chang, and Z. Tu. Multiple clustered instance learning for histopathology cancer image classification, segmentation and clustering. In *CVPR*, 2012.
- [30] Q. Zhang, S.A. Goldman, et al. EM-DD: An improved multiple-instance learning technique. *NIPS*, 14, 2001.
- [31] Z.H. Zhou, Y.Y. Sun, and Y.F. Li. Multi-instance learning by treating instances as non-iid samples. In *ICML*, 2009.
- [32] Z.H. Zhou, X.B. Xue, and Y. Jiang. Locating regions of interest in cbir with multi-instance learning techniques. In *AI 2005: Advances in Artificial Intelligence*, pages 92–101. Springer, 2005.

Closed-form Solutions to a Subclass of Continuous Stochastic Games via Symbolic Dynamic Programming

Shamin Kinathil
ANU and NICTA
Canberra, ACT, Australia
shamin.kinathil@anu.edu.au

Scott Sanner
NICTA and ANU
Canberra, ACT, Australia
ssanner@nicta.com.au

Nicolás Della Penna
ANU and NICTA
Canberra, ACT, Australia
nicolas.della-penna@anu.edu.au

Abstract

Zero-sum stochastic games provide a formalism to study competitive sequential interactions between two agents with diametrically opposing goals and evolving state. A solution to such games with discrete state was presented by Littman (Littman, 1994). The continuous state version of this game remains unsolved. In many instances continuous state solutions require non-linear optimisation, a problem for which closed-form solutions are generally unavailable. We present an exact closed-form solution to a subclass of zero-sum continuous stochastic games that can be solved as a parameterised linear program by utilising symbolic dynamic programming. This novel technique is applied to calculate exact solutions to a variety of zero-sum continuous state stochastic games.

1 INTRODUCTION

Modelling competitive sequential interactions between agents has important applications within economic and financial decision-making. Stochastic games (Shapley, 1953) provide a convenient framework to model sequential interactions between non-cooperative agents. In zero-sum stochastic games, participating agents have diametrically opposing goals. A reinforcement learning solution to discrete state zero-sum stochastic games was presented by Littman (Littman, 1994). Closed-form solutions for the continuous state case remain unknown, despite the general importance of this formalism — zero-sum continuous state stochastic games provide a convenient framework with which to model robust sequential optimisation in adversarial settings including domains such as option valuation on derivative markets.

The difficulty of solving zero-sum continuous state stochastic games originates from the need to calculate a

Nash equilibrium for every state, of which there are infinitely many. In this paper we make the following key contributions:

- We characterise a subclass of zero-sum continuous state stochastic games with restricted reward and transition functions that can be solved exactly via parameterised linear optimisation.
- We provide an algorithm that solves this subclass of stochastic games exactly and optimally using Symbolic Dynamic Programming (SDP) (Boutilier *et al.*, 2001; Sanner *et al.*, 2011; Zamani & Sanner, 2012) for arbitrary horizons.

This paper is organised as follows: In Section 2 we describe Markov Decision Processes (MDPs) (Howard, 1960) and value iteration (Bellman, 1957), a widely used dynamic programming method for solving MDPs. In Sections 3 and 4, we present zero-sum stochastic games with discrete and continuous states, respectively, as game-theoretic generalisations of the MDP framework. Following this, in Section 5, we introduce SDP, and show how it can be used to calculate the first known exact solution to a particular subclass of zero-sum continuous state stochastic games. In Section 6 we calculate exact solutions to three empirical domains: a continuous state generalisation of matching pennies, binary option valuation and robust energy production. In Section 7, we survey the related literature. We conclude in Section 8 and identify interesting directions for future research.

2 MARKOV DECISION PROCESSES

A Markov Decision Process (MDP) (Howard, 1960) is defined by the tuple $\langle S, A, T, R, h, \gamma \rangle$. S and A specify a finite set of states and actions, respectively. T is the transition function $T : S \times A \rightarrow S$, which defines the effect of an action on the state. R is the reward function $R : S \times A \rightarrow \mathbb{R}$, which encodes the preferences of the agent. The horizon h represents the number of decision

steps until termination and the discount factor $\gamma \in [0, 1]$ is used to discount future rewards. In general, an agent's objective is to find a policy, $\pi : S \rightarrow A$, which maximises the expected sum of discounted rewards over horizon h .

Value iteration (VI) (Bellman, 1957) is a general dynamic programming algorithm used to solve MDPs. VI is based on the set of Bellman equations, which mathematically express the optimal solution of an MDP. They provide a recursive expansion to compute: (1) $V^*(s)$, the expected value of following the optimal policy in state s ; and (2) $Q^*(s, a)$, the expected quality of taking a in state s , then following the optimal policy. The key idea of VI is to successively approximate $V^*(s)$ and $Q^*(s, a)$ by $V^h(s)$ and $Q^h(s, a)$, respectively, at each horizon h . These two functions satisfy the following recursive relationship:

$$Q^h(s, a) = R(s, a) + \gamma \cdot \sum_{s' \in S} T(s, a, s') \cdot V^{h-1}(s') \quad (1)$$

$$V^h(s) = \max_{a \in A} \{Q^h(s, a)\} \quad (2)$$

The algorithm can be executed by first initialising $V^0(s)$ to zero or the terminal reward. Then for each h , $V^h(s)$ is calculated from $V^{h-1}(s)$ via Equations (1) and (2), until the intended h -stage-to-go value function is computed. Value iteration converges linearly in the number of iterations to the true values of $Q^*(s, a)$ and $V^*(s)$ (Bertsekas, 1987).

MDPs can be used to model multiagent systems by assuming that other agents are part of the environment and have fixed behaviour. As a result, they ignore the difference between responsive agents and a passive environment (Hu & Wellman, 1998). In the next two sections we present game theoretic frameworks which generalises MDPs to situations with two or more responsive agents.

3 ZERO-SUM DISCRETE STOCHASTIC GAMES

Discrete state stochastic games (DSGs) are formally defined by the tuple $\langle S, A_1, \dots, A_n, T, R_1, \dots, R_n, h, \gamma \rangle$. S is a set of discrete states and A_i is the action set available to agent i . T is a transition function $T : S \times A_1 \times \dots \times A_n \rightarrow \Delta(S)$, where $\Delta(S)$ is the set of probability distributions over the state space S . The reward function $R_i : S \times A_1 \times \dots \times A_n \rightarrow \mathbb{R}$, encodes the individual preferences of agent i . The horizon h represents the number of decision steps until termination and the discount factor $\gamma \in [0, 1]$ is used to discount future rewards. In general, an agent's objective is to find a policy, $\pi_i : S \rightarrow \sigma_i(A_i)$ which maximises the expected sum of discounted rewards over horizon h . Here $\sigma_i(A_i)$ specifies probability distributions over action set A_i . The optimal policy in a DSG may be stochastic, that is, it may define a mixed strategy for each state.

Zero-sum DSGs are a type of DSG involving two agents with diametrically opposing goals. Under these conditions the reward structure for the game can be represented by a single reward function since an agent's reward function is simply the opposite of their opponent's. The objective of each agent is to maximise its expected discounted future rewards in the minimax sense. That is, each agent views its opponent as acting to minimise the agent's reward. Zero-sum DSGs can be solved using a technique analogous to value iteration for MDPs (Littman, 1994). The value function, $V^h(s)$, in this setting can be defined as:

$$V^h(s) = \max_{m \in \sigma_1(A_1)} \min_{o \in \sigma_2(A_2)} \sum_{a_1 \in A_1} \sum_{a_2 \in A_2} Q^h(s, a_1, a_2) \cdot m_{a_1} \cdot o_{a_2} \quad (3)$$

where $m \in \mathbb{R}^{|A_1|}$ and $o \in \mathbb{R}^{|A_2|}$ are mixed (stochastic) strategies from $\sigma_1(A_1)$ and $\sigma_2(A_2)$, respectively. $Q^h(s, a_1, a_2)$, the quality of taking action a_1 against action a_2 in state s , is given by:

$$Q^h(s, a_1, a_2) = R(s, a_1, a_2) + \gamma \cdot \sum_{s' \in S} T(s, a_1, a_2, s') \cdot V^{h-1}(s'). \quad (4)$$

Equation (3) can be further simplified by noting that given any m , the optimal minimum strategy is achieved through a deterministic action choice. This observation leads to the following form:

$$V^h(s) = \max_{m \in \sigma_1(A_1)} \min_{a_2 \in A_2} \sum_{a_1 \in A_1} Q^h(s, a_1, a_2) \cdot m_{a_1}. \quad (5)$$

Together Equations (4) and (5) define a recursive method to calculate the optimal solution to zero-sum DSGs. The policy for the opponent can be calculated by applying symmetric reasoning and the Minimax theorem (Neumann, 1928).

3.1 SOLUTION TECHNIQUES

Zero-sum DSGs can be solved via discrete linear optimisation at each horizon h . The value function in Equation (5) can be reformulated as a linear program through the following steps:

1. Define $V^h(s)$ to be the value of the inner minimisation term in Equation (5). This leads to the following linear program for a known state s :

$$\begin{aligned} & \text{maximise } V^h(s) \\ & \text{subject to} \\ & V^h(s) = \min_{a_2 \in A_2} \sum_{a_1 \in A_1} Q^h(s, a_1, a_2) \cdot m_{a_1} \quad (6a) \\ & \sum_{a_1 \in A_1} m_{a_1} = 1; m_{a_1} \geq 0 \quad \forall a_1 \in A_1 \end{aligned}$$

2. Replace the equality ($=$) in constraint (6a) with \leq by observing that the maximisation of $V^h(s)$ effectively pushes the \leq condition to the $=$ case. This gives:

$$\begin{aligned} & \text{maximise } V^h(s) \\ & \text{subject to} \\ & V^h(s) \leq \min_{a_2 \in A_2} \sum_{a_1 \in A_1} Q^h(s, a_1, a_2) \cdot m_{a_1} \quad (7a) \\ & \sum_{a_1 \in A_1} m_{a_1} = 1; m_{a_1} \geq 0 \quad \forall a_1 \in A_1 \end{aligned}$$

3. Remove the minimisation operator in constraint (7a) by noting that the minimum of a set is less than or equal to the minimum of all elements in the set. This leads to the final form of the discrete linear optimisation problem:

$$\begin{aligned} & \text{maximise } V^h(s) \\ & \text{subject to} \\ & V^h(s) \leq \sum_{a_1 \in A_1} Q^h(s, a_1, a_2) \cdot m_{a_1} \quad \forall a_2 \in A_2 \\ & \sum_{a_1 \in A_1} m_{a_1} = 1; m_{a_1} \geq 0 \quad \forall a_1 \in A_1 \end{aligned}$$

We can now use existing linear programming solvers to compute the optimal solution to this linear program for each $s \in S$ at a given horizon h .

The linear program used to solve zero-sum DSGs cannot be used with continuous state formulations, since there are infinitely many states. A key contribution of this paper is in showing that zero-sum continuous state stochastic games can still be solved exactly through the use of symbolic dynamic programming. In the next section we present the continuous state analogue to zero-sum DSGs.

4 ZERO-SUM CONTINUOUS STOCHASTIC GAMES

Continuous state stochastic games (CSGs) are formally defined by the tuple $\langle \vec{x}, A_1, \dots, A_n, T, R_1, \dots, R_n, h, \gamma \rangle$. In CSGs states are represented by vectors of continuous variables, $\vec{x} = (x_1, \dots, x_m)$, where $x_i \in \mathbb{R}$. The other components of the tuple are as previously defined in Section 3.

Zero-sum CSGs impose the same restrictions on the number of agents and the reward structure as their discrete state counterparts.

The optimal solution to zero-sum CSGs can be calculated via the following recursive functions:

$$Q^h(\vec{x}, a_1, a_2) = R(\vec{x}, a_1, a_2) + \gamma \cdot \int T(\vec{x}, a_1, a_2, \vec{x}') \cdot V^{h-1}(\vec{x}') d\vec{x}' \quad (8)$$

$$V^h(\vec{x}) = \max_{m \in \sigma(A_1)} \min_{a_2 \in A_2} \sum_{a_1 \in A_1} Q^h(\vec{x}, a_1, a_2) \cdot m_{a_1} \quad (9)$$

We can derive Equation (8) from Equation (4) by replacing s, s' and the \sum operator with their continuous state counterparts, \vec{x}, \vec{x}' and \int , respectively. Equation (9) is simply Equation (5) restated.

4.1 SOLUTION TECHNIQUES

Zero-sum CSGs can be solved using a technique analogous to that presented in Section 3.1. Namely, the value function in Equation (9) can be reformulated as the following continuous optimisation problem:

$$\begin{aligned} & \text{maximise } V^h(\vec{x}) \\ & \text{subject to} \\ & V^h(\vec{x}) \leq \sum_{a_1 \in A_1} Q^h(\vec{x}, a_1, a_2) \cdot m_{a_1} \quad \forall a_2 \in A_2 \quad (10a) \\ & \sum_{a_1 \in A_1} m_{a_1} = 1; m_{a_1} \geq 0 \quad \forall a_1 \in A_1 \end{aligned}$$

This optimisation problem cannot be easily solved using existing techniques due to two factors: (1) there are infinitely many states in \vec{x} ; and (2) constraint (10a) is nonlinear in \vec{x} and m_{a_1} for general representations of $Q^h(\vec{x}, a_1, a_2)$. To further illustrate the second limitation consider $Q^h(\vec{x}, a_1, a_2)$ in the form of a linear function in x , for some a_1 and a_2 :

$$Q^h(\vec{x}, a_1, a_2) = \sum_j c_j \cdot x_j \quad (11)$$

Substituting Equation (11) into constraint (10a) yields:

$$V^h(\vec{x}) \leq \sum_{a_1 \in A_1} m_{a_1} \sum_j c_j \cdot x_j \quad \forall a_2 \in A_2. \quad (12)$$

It is clear from Equation (12) that a linear representation of $Q^h(\vec{x}, a_1, a_2)$ leads to a nonlinear constraint where m_{a_1} must be optimal with respect to the free variable \vec{x} . This results in a parameterised nonlinear program, whose optimal solutions are known to be NP-hard (Bennett & Mangasarian, 1993; Petrik & Zilberstein, 2011).

At this point we present the first key insight of this paper: we can transform constraint (10a) from a parameterised nonlinear constraint to a piecewise linear constraint by imposing the following restrictions: (1) restricting the reward function, $R(\vec{x}, a_1, a_2)$, to piecewise constant functions; and (2) restricting the transition function, $T(\vec{x}, a_1, a_2, \vec{x}')$, to piecewise linear functions. As a result, $V^h(\vec{x})$ and $Q^h(\vec{x}, a_1, a_2)$ will be piecewise constant functions, thereby guaranteeing a tractable solution to constraint (10a).

One key challenge still remains, namely, dealing with the infinitely many states in \vec{x} . We know that the $V^h(\vec{x})$ and $Q^h(\vec{x}, a_1, a_2)$ functions have structure, but are unable to derive them. Furthermore, given known structures for $V^h(\vec{x})$ and $Q^h(\vec{x}, a_1, a_2)$ we must determine the restrictions that

guarantee a tractable solution. The SDP framework in conjunction with its closed-form operations provide answers to both of these concerns.

In the next section we show that zero-sum CSGs, with the aforementioned restrictions, can be solved optimally for arbitrary horizons using symbolic dynamic programming.

5 SYMBOLIC DYNAMIC PROGRAMMING

Symbolic dynamic programming (SDP) (Boutillier *et al.*, 2001) is the process of performing dynamic programming via symbolic manipulation. In the following sections we present a brief overview of SDP operations and also show how SDP can be used to solve zero-sum CSGs.

5.1 CASE REPRESENTATION

SDP assumes that all functions can be represented in case statement form (Boutillier *et al.*, 2001) as follows:

$$f = \begin{cases} \phi_1 : f_1 \\ \vdots \\ \phi_k : f_k \end{cases}$$

Here, the ϕ_i are logical formulae defined over the state \vec{x} that can consist of arbitrary logical combinations of boolean variables and linear inequalities ($\geq, >, <, \leq$) over continuous variables. We assume that the set of conditions $\{\phi_1, \dots, \phi_k\}$ disjointly and exhaustively partition \vec{x} such that f is well-defined for all \vec{x} . In general, the f_i may be polynomials of \vec{x} with non-negative exponents. However, in this paper we restrict the f_i to be either constant or linear functions of the state variable \vec{x} . Henceforth, we refer to functions with linear ϕ_i and piecewise constant f_i as linear piecewise constant (LPWC) and functions with linear ϕ_i and piecewise linear f_i as linear piecewise linear (LPWL) functions.

5.2 CASE OPERATIONS

Operations on case statements may be either unary or binary. In this section we present a brief overview of the SDP operations needed to calculate closed form solutions to zero-sum CSGs. All of the operations presented here are closed form for LPWC and LPWL functions. We refer the reader to (Sanner *et al.*, 2011; Zamani & Sanner, 2012) for more thorough expositions of SDP and its operations.

Unary operations on a single case statement f , such as scalar multiplication $c \cdot f$ where $c \in \mathbb{R}$, are applied to each f_i ($1 \leq i \leq k$).

Binary operations such as addition, subtraction and multiplication are executed in two stages. Firstly, the cross-

product of the logical partitions of each case statement is taken, producing paired partitions. Finally, the binary operation is applied to the resulting paired partitions. The “cross-sum” \oplus operation can be performed on two cases in the following manner:

$$\begin{cases} \phi_1 : f_1 \\ \phi_2 : f_2 \end{cases} \oplus \begin{cases} \psi_1 : g_1 \\ \psi_2 : g_2 \end{cases} = \begin{cases} \phi_1 \wedge \psi_1 : f_1 + g_1 \\ \phi_1 \wedge \psi_2 : f_1 + g_2 \\ \phi_2 \wedge \psi_1 : f_2 + g_1 \\ \phi_2 \wedge \psi_2 : f_2 + g_2 \end{cases}$$

“cross-subtraction” \ominus and “cross-multiplication” \otimes are defined in a similar manner but with the addition operator replaced by the subtraction and multiplication operators, respectively. Some partitions resulting from case operators may be inconsistent and are thus removed.

Minimisation over cases, known as *casemin*, is defined as:

$$\text{casemin} \left(\begin{cases} \phi_1 : f_1 \\ \phi_2 : f_2 \end{cases}, \begin{cases} \psi_1 : g_1 \\ \psi_2 : g_2 \end{cases} \right) = \begin{cases} \phi_1 \wedge \psi_1 \wedge f_1 < g_1 : f_1 \\ \phi_1 \wedge \psi_1 \wedge f_1 \geq g_1 : g_1 \\ \phi_1 \wedge \psi_2 \wedge f_1 < g_2 : f_1 \\ \phi_1 \wedge \psi_2 \wedge f_1 \geq g_2 : g_2 \\ \vdots \\ \vdots \end{cases}$$

casemin preserves the linearity of the constraints and the constant or linear nature of the f_i and g_i . If the f_i or g_i are quadratic then the expressions $f_i > g_i$ or $f_i \leq g_i$ will be at most univariate quadratic and any such constraint can be linearised into a combination of at most two linear inequalities by completing the square.

Substitution into case statements is performed via a set θ of variables and their substitutions e.g. $\theta = \{x'_1/(x_1 + x_2), x'_2/x_1^2 \exp(x_2)\}$, where the LHS of the $/$ represents the substitution variable and the RHS of the $/$ represents the expression that should be substituted in its place. θ can be applied to both non-case functions f_i and case partitions ϕ_i as $f_i\theta$ and $\phi_i\theta$, respectively. Substitution into case statements can be written as:

$$f\theta = \begin{cases} \phi_1\theta : f_1\theta \\ \vdots \\ \phi_k\theta : f_k\theta \end{cases}$$

Substitution is used when calculating integrals with respect to deterministic δ transitions (Sanner *et al.*, 2011).

A case statement can be maximised with respect to a continuous parameter y as $f_1(\vec{x}, y) = \max_y f_2(\vec{x}, y)$. The continuous maximisation operation is a complex case operation whose explanation is beyond the scope of this paper. We refer the reader to (Zamani & Sanner, 2012) for further details.

Case statements and their operations are implemented using Extended Algebraic Decision Diagrams (XADDs) (Sanner *et al.*, 2011). XADDs provide a compact

data structure with which to maintain compact forms of $Q^h(\vec{x}, a_1, a_2)$ and $V^h(\vec{x})$.

5.3 SDP FOR ZERO-SUM CONTINUOUS STOCHASTIC GAMES

In this section we will show that a subclass of zero-sum continuous stochastic games with (a) piecewise constant rewards; and (b) piecewise linear transition functions can be solved exactly and in closed-form by using SDP.

To calculate the exact solution to zero-sum CSGs we begin by replacing all functions and operations in Equations (8) and (9) by their case statement equivalents. That is, we exchange operations such as $+$, \times and \min , by their symbolic equivalents, \oplus , \otimes and casemin , respectively, and express $R(\vec{x}, a_1, a_2)$, $T(\vec{x}, a_1, a_2, \vec{x}')$, $V^0(\vec{x})$ as case statements. m_{a_1} is also encoded as a trivial case statement representing an uninstantiated symbolic variable:

$$m_{a_1} = \left\{ \top : m_{a_1} \right.$$

The optimal solution to zero-sum CSGs can now be described by the following recursive SDP equations:

$$Q^h(\vec{x}, a_1, a_2) = R(\vec{x}, a_1, a_2) \oplus \gamma \otimes \int T(\vec{x}, a_1, a_2, \vec{x}') \otimes V^{h-1}(\vec{x}') d\vec{x}' \quad (13)$$

$$\tilde{Q}^h(\vec{x}, a_2) = \sum_{a_1 \in A_1} Q^h(\vec{x}, a_1, a_2) \otimes m_{a_1} \quad (14)$$

$$V^h(\vec{x}) = \max_m \text{casemin} \left(\text{casemin}_{a_2 \in A_2} \left(\tilde{Q}^h(\vec{x}, a_2) \right), \mathbb{I} \right) \quad (15)$$

Equation (14) calculates a symbolic Q function weighted by the variable m_{a_1} for each a_1 . In Equation (15) the inner casemin operation is calculated with respect to $\tilde{Q}^h(\vec{x}, a_2)$ instantiated with a particular a_2 . The “indicator” function \mathbb{I} serves as the summation constraint $\sum_{a_1 \in A_1} m_{a_1} = 1$ and ensures that the subsequent \max operation returns legal values for the m_{a_1} . The indicator is defined as follows:

$$\mathbb{I} = \begin{cases} \forall a_1 \in A_1 \quad [(m_{a_1} \geq 0) \wedge (m_{a_1} \leq 1) \wedge (\sum m_{a_1} = 1)] : +\infty \\ \forall a_1 \in A_1 \neg [(m_{a_1} \geq 0) \wedge (m_{a_1} \leq 1) \wedge (\sum m_{a_1} = 1)] : -\infty \end{cases}$$

The function \mathbb{I} returns $+\infty$ when the conjunction of all constraints on each m_{a_1} are satisfied and $-\infty$, otherwise.

In Algorithm 1 we present CSG-VI, a methodology to calculate the optimal h -stage-to-go value function through Equations (13) to (15). In the algorithm we notationally specify the arguments of the Q^h and V^h functions when they are instantiated by an operation. For the base case of $h = 0$, we set $V^0(\vec{x})$, expressed in case statement form, to zero or to the terminal reward. For all $h > 0$ we apply the

previously defined SDP operations in the following steps:

1. Prime the Value Function. In line 6 we set up a substitution $\theta = \{x_1/x'_1, \dots, x_m/x'_m\}$, and obtain $V^{h'} = V^h\theta$, the “next state”.
2. Discount and add Rewards. We assume that the reward function contains primed variables and incorporate it in line 8.
3. Continuous Regression. For the continuous state variables in \vec{x} lines 10 – 11 follow the rules of integration w.r.t. a δ function (Sanner *et al.*, 2011). This yields the following symbolic substitution: $\int f(x'_j) \otimes \delta[x'_j - g(\vec{z})] dx'_j = f(x'_j) \{x'_j/g(\vec{z})\}$, where $g(\vec{z})$ is a case statement and \vec{z} does not contain x'_j . The latter operation indicates that any occurrence of x'_j in $f(x'_j)$ is symbolically substituted with the case statement $g(\vec{z})$.
4. Incorporate Agent 1’s strategy. At this point we have calculated $Q^h(\vec{x}, a_1, a_2)$, as shown in Equation (13). In lines 13 - 14 we weight the strategy for a specific a_1 by m_{a_1} . We note that m_{a_1} is a case statement representing an uninstantiated symbolic variable.
5. Case Minimisation. In lines 16 – 17 we compute the best case for a_2 as a function of all other variables, as shown in Equation (14).
6. Incorporate Constraints. In line 19 we incorporate constraints on the m_{a_1} variable: $\sum_{a_1 \in A_1} m_{a_1} = 1$ and $m_{a_1} \geq 0 \wedge m_{a_1} \leq 1 \quad \forall a_1 \in A_1$. The casemin operator ensures that all case partitions which involve illegal values of m_{a_1} are mapped to $-\infty$.
7. Maximisation. In lines 22 – 23 we compute the best response strategy for every state. We note that this can only be done via symbolic methods since there are infinitely many states. At this point we have calculated $V^h(\vec{x})$ as shown in Equation (15).

It can be proved that all of the SDP operations used in Algorithm 1 are closed form for LPWC or LPWL functions (Sanner *et al.*, 2011; Zamani & Sanner, 2012). Given a LPWC $V^0(\vec{x})$ and that SDP operations are closed form, the resulting $V^{h+1}(\vec{x})$ is also LPWC. Therefore, by induction $V^{h+1}(\vec{x})$ is LPWC for all horizons h .

In the next section we demonstrate how SDP can be used to compute exact solutions to a variety of zero-sum continuous stochastic games.

6 EMPIRICAL RESULTS

In this section we evaluate our novel SDP solution technique for zero-sum CSGs on three continuous domains¹:

¹All of the source code can be found online at <http://code.google.com/p/xadd-inference>.

Algorithm 1: CSG-VI(CSG, H, \mathbb{I}) $\rightarrow (V^h)$

```

1 begin
2    $V^0 := 0, h := 0$ 
3   while  $h < H$  do
4      $h := h + 1$ 
5     // Prime the value function
6      $Q^h := \text{Prime}(V^{h-1})$ 
7     // Discount and add Rewards
8      $Q^h := R(\vec{x}, a_1, a_2, \vec{x}') \oplus (\gamma \otimes Q^h)$ 
9     // Continuous Regression
10    for all  $x'_j \in Q^h$  do
11       $| Q^h := \int Q^h \otimes T(x'_j | a_1, a_2, \vec{x}) d_{x'_j}$ 
12    // Incorporate Agent 1's strategy
13    for all  $a_1 \in A_1$  do
14       $| Q^h := Q^h \oplus (Q^h(a_1) \otimes \{\top : m_{a_1}\})$ 
15    // Case Minimisation
16    for all  $a_2 \in A_2$  do
17       $| Q^h := \text{casemin}(Q^h, Q^h(a_2))$ 
18    // Incorporate constraints
19     $Q^h := \text{casemin}(Q^h, \mathbb{I})$ 
20    // Maximisation
21     $V^h = Q^h$ 
22    for all  $a_1 \in A_1$  do
23       $| V^h := \max_{m_{a_1}} V^h(m_{a_1})$ 
24    // Terminate if early convergence
25    if  $V^h = V^{h-1}$  then
26       $| \text{break}$ 
27  return  $(V^h)$ 

```

(1) continuous stochastic matching pennies; (2) binary option stochastic game; and (3) robust energy production. The results represent the first known exact solutions to these domains.

6.1 CONTINUOUS STOCHASTIC MATCHING PENNIES

Matching pennies is a well known zero-sum game with a mixed strategy Nash Equilibrium (Osborne, 2004). In this paper we extend the standard formulation of the game by incorporating continuous state and sequential decisions while still maintaining the zero-sum nature of the reward.

6.1.1 Domain Description

We define continuous stochastic matching pennies as an extensive form game between two players $p \in \{1, 2\}$. The aim of a player is to maximise its expected discounted pay-off at a fixed horizon H . Our game is played within the interval $[0, 1]$, two fixed variables $c \in [0, 1)$ and $d \in (0, 1]$ with $(c < d)$, are used to partition the interval into three regions $r \in \{1, 2, 3\}$. Each region is associated with its own

zero-sum reward structure. The continuous state variable $x \in [0, 1]$ is used to specify which region the players are competing within.

At each horizon ($h \leq H$) each player executes an action $a_p \in \{\text{heads}_p, \text{tails}_p\}$. Player 1 “wins” if both players choose the same action. Otherwise, Player 2 wins. The joint actions of the players affect the state x as follows:

$$P(x' | x, a_1, a_2) = \delta \left[x' - \begin{cases} (\text{heads}_1) \wedge (\text{heads}_2) \wedge (x \geq k) : & x - k \\ (\text{heads}_1) \wedge (\text{tails}_2) \wedge (x \leq 1) : & x + k \\ (\text{tails}_1) \wedge (\text{heads}_2) \wedge (x \geq k) : & x + k \\ (\text{tails}_1) \wedge (\text{tails}_2) \wedge (x \leq 1) : & x - k \end{cases} \right]$$

The constant $k \in (0, 1)$ is a step size which perturbs the state x . If Player 1 wins, the state moves to the left by k , otherwise it moves to the right by k . The Dirac function $\delta[\cdot]$ ensures that the transitions are valid conditional probability functions that integrate to 1.

We define the rewards obtained by Player 1 in region r as:

$$R_1^r = \begin{cases} (\text{heads}_1) \wedge (\text{heads}_2) : & \alpha_1^r \\ (\text{heads}_1) \wedge (\text{tails}_2) : & \alpha_2^r \\ (\text{tails}_1) \wedge (\text{heads}_2) : & \alpha_3^r \\ (\text{tails}_1) \wedge (\text{tails}_2) : & \alpha_4^r \end{cases}$$

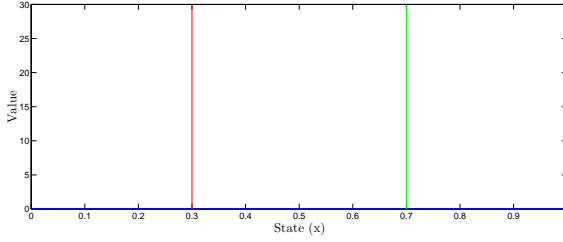
Here we restrict $\alpha_i^r \in \mathbb{R}$. The rewards obtained by Player 2 in the same region are simply $-R_1^r$. Given this reward formulation we specify two different reward structures: symmetric and asymmetric. In a symmetric reward structure $\alpha_1^r = \alpha_4^r$ and $\alpha_2^r = \alpha_3^r$. An example of this reward structure is shown in Table 1. Under a symmetric reward setting the expected reward for each player is the same across all regions r . In an asymmetric reward structure we allow each of the α_i^r to differ in both sign and magnitude. Table 2 shows an example of an asymmetric reward structure. Under an asymmetric setting the expected reward for each player may vary across each region r . This gives a player an incentive to reach regions with a higher expected reward.

Table 1: Symmetric reward structure for Player 1.

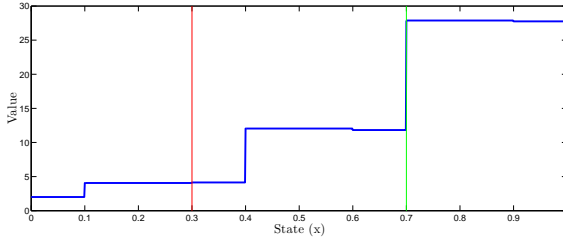
	Region 1	Region 2	Region 3
$(\text{heads}_1) \wedge (\text{heads}_2)$	10	5	20
$(\text{heads}_1) \wedge (\text{tails}_2)$	-10	-5	-20
$(\text{tails}_1) \wedge (\text{heads}_2)$	-10	-5	-20
$(\text{tails}_1) \wedge (\text{tails}_2)$	10	5	20

Table 2: Asymmetric reward structure for Player 1.

	Region 1	Region 2	Region 3
$(\text{heads}_1) \wedge (\text{heads}_2)$	1	5	7
$(\text{heads}_1) \wedge (\text{tails}_2)$	-3	-5	-2
$(\text{tails}_1) \wedge (\text{heads}_2)$	0	-5	10
$(\text{tails}_1) \wedge (\text{tails}_2)$	2	5	20



(a) Symmetric rewards.



(b) Asymmetric rewards.

Figure 1: The optimal value functions of the continuous stochastic matching pennies game for Player 1 at horizon 4 under (a) symmetric and (b) asymmetric reward structures. Threshold values are set to $c = 0.3$ and $d = 0.7$ and are highlighted in red and green, respectively. The step size is $k = 0.3$.

6.1.2 Results

We investigate the continuous stochastic matching pennies game under both symmetric and asymmetric reward structures. For both experiments the threshold values are set to $c = 0.3$ and $d = 0.7$. The step size is $k = 0.3$.

Figure (1a) shows the results of the continuous stochastic matching pennies game using the symmetric reward structure given in Table 1. The results show that the expected reward for Player 1 remains at zero over all 4 horizons, irrespective of the state x . The symmetric reward structure clearly shows that both players achieve the same expected reward in all regions r . This in turn ensures that both players are indifferent between their pure strategies. Hence, the expected reward for each player is zero in all regions. This result corresponds to the well known solution of the matching pennies game with symmetric rewards and serves as a proof of concept for our novel solution technique.

Figure (1b) shows the effect of the asymmetric reward structure given in Table 2. The figure shows that Player 1 achieves the highest expected reward in Region 3, followed by Region 2 and finally by Region 1. This corresponds to the expected reward within each region of Table 2. The results indicate that the Player 1 is no longer indifferent between its pure strategies in each region and may take short-term losses to reach more favourable regions.

6.2 BINARY OPTION STOCHASTIC GAME

Binary options are financial instruments which allow an investor to bet on the outcome of a yes/no proposition. The proposition typically relates to whether the price of a particular asset that underlies the option will rise above or fall below a specified amount, known as the strike price, $\kappa \in \mathbb{R}$. When the option reaches maturity the investor receives a fixed pay-off if their bet was correct and nothing otherwise.

6.2.1 Domain Description

We analyse the valuation of a binary option as an extensive form zero-sum game between a trader and the market. The aim of the trader is to maximise their expected discounted pay-off at a fixed horizon H through buying and selling options within an adversarial market. The problem has two state variables: the underlying market value of the asset $v \in [0, 100]$ and the trader's inventory of options $i \in \mathbb{N}$.

At each time step the trader can execute one of three actions $a_{trd} \in \{buy_{trd}, sell_{trd}, hold_{trd}\}$, where buy_{trd} refers to a request to buy an option from the market, $sell_{trd}$ refers to a request to sell an option to the market and $hold_{trd}$ is equivalent to taking no action. The market can execute one of two actions: $a_{mkt} \in \{sell_{mkt}, nsell_{mkt}\}$, where $sell_{mkt}$ corresponds to selling an option to the trader and $nsell_{mkt}$ corresponds to not selling to the trader.

The joint actions of the trader and market, a_{trd} and a_{mkt} , respectively, affect both the underlying market value of the asset and the trader's inventory. For the sake of simplicity we assume that the market value may increase or decrease by fixed step sizes, $u \in \mathbb{R}$ for an increase and $d \in \mathbb{R}$ for a decrease.

The trader's option inventory dynamics are given by:

$$P(i'|v, i, a_{trd}, a_{mkt}) = \delta \left[i' - \begin{cases} (buy_{trd}) \wedge (sell_{mkt}) : & i + 1 \\ (sell_{trd}) \wedge (i > 0) : & i - 1 \\ otherwise : & i \end{cases} \right]$$

It should be noted that under this formulation the market will always buy an option from the trader when the trader selects $sell_{trd}$. The market value changes according to:

$$P(v'|v, i, a_{trd}, a_{mkt}) = \delta \left[v' - \begin{cases} (buy_{trd}) \wedge (sell_{mkt}) : & v + u \\ (sell_{trd}) \wedge (i > 0) : & v - d \\ otherwise : & v \end{cases} \right]$$

Assuming that the strike price $\kappa \in [0, 100]$, the rewards obtained by the trader are given by:

$$R_{trader} = \begin{cases} (sell_{trd}) \wedge (i > 0) \wedge (v > \kappa) : & 1 \\ otherwise : & 0 \end{cases}$$

The market's reward is simply the additive inverse of the trader's reward. Hence, the binary option game is zero-sum.

6.2.2 Results

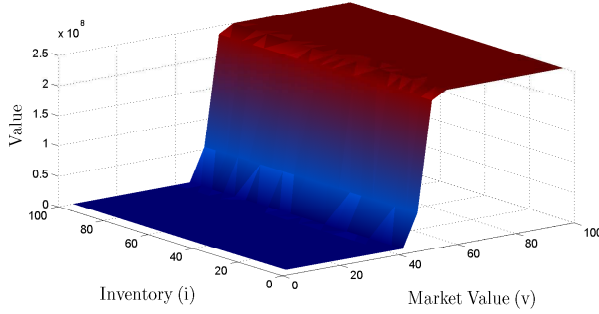


Figure 2: The optimal value function of the binary option stochastic game for the trader at horizon 20. The strike price is set to $\kappa = 45.0$ and the increment and decrement values are set to $u = 1.0$ and $d = 1.0$, respectively. Under the domain specification the value function is invariant to the inventory i .

In Figure (2) we show the optimal value function for the binary option game at horizon 20. The strike price is set to $\kappa = 45.0$ and the increment and decrement values, u and d are both set to 1.0. The value function clearly shows that under this formulation the trader achieves the most reward by selling options as soon $v > \kappa$. Selling an option causes the underlying value to decrease. Once the value falls beneath the strike price, the trader will buy options, which increases the underlying value. This leads to the continual cycling of buying and selling of the option at values close to the strike price κ . In essence the trader behaves like a market maker in that they take both sides of the transaction at values near κ . We note that while Figure (2) is invariant to the inventory of options i , its inclusion is critical for the correct formalisation of the domain.

6.3 ROBUST ENERGY PRODUCTION

The provision of energy resources is an integral component of any economy. Energy providers must be able to produce energy in response to changes in energy demand. In situations where demand exceeds supply, an energy crisis may occur. In this paper we investigate energy production from the viewpoint of an energy provider responsible for supplying energy in an adversarial environment.

6.3.1 Domain Description

We define our energy production domain as an extensive form zero-sum game between an energy provider and nature. The aim of the energy provider is to maximise its

expected discounted reward at a fixed horizon H by changing production levels in response to changes in demand. The domain has two state variables: the production level $p \in \mathbb{R}^+$ and the energy demand $d \in \mathbb{R}^+$.

At each time step the energy provider can execute one of two actions $a_{prd} \in \{inc_{prd}, dec_{prd}\}$, where inc_{prd} refers to increasing energy production and dec_{prd} refers to decreasing energy production. Nature can also execute one of two actions $a_{nat} \in \{inc_{dem}, dec_{dem}\}$, where inc_{dem} refers to increasing energy demand and dec_{dem} refers to decreasing energy demand. We specify the increase in the amount of energy produced or demanded by $prd_u, nat_u \in \mathbb{R}^+$ and a corresponding decrease by $prd_d, nat_d \in \mathbb{R}^+$.

The joint actions of the energy provider and nature, a_{prd} and a_{nat} , respectively, affect the production level as follows:

$$P(p'|d, p, a_{prd}, a_{nat}) = \delta \left[p' - \begin{cases} (inc_{prd}) : & p + prd_u \\ (dec_{prd}) \wedge (p > prd_d) : & p - prd_d \\ otherwise : & p \end{cases} \right]$$

The energy demand changes according to:

$$P(d'|d, p, a_{prd}, a_{nat}) = \delta \left[d' - \begin{cases} (inc_{dem}) : & d + nat_u \\ (dec_{dem}) \wedge (d > nat_d) : & d - nat_d \\ otherwise : & d \end{cases} \right]$$

The reward obtained by the energy provider are specified as:

$$R_{prd} = \begin{cases} (p < d) : & -100 \\ otherwise : & 0 \end{cases}$$

We note that under this reward structure failure to meet energy demand is heavily penalised, whereas meeting or even exceeding demand are given the same reward. Nature's reward is simply the additive inverse of the energy provider's reward.

6.3.2 Results

In Figure (3) we show the optimal value function for the robust energy production game at horizon 8. The production and demand increase and decrease variables were set to $prd_u = prd_d = 1.0$ and $nat_u = nat_d = 0.5$, respectively. The value function shows that the energy provider achieves the highest value when the energy provided meets or exceeds demand. The value function is lowest when the demand exceeds supply, which is in accordance with the reward structure. The value function clearly decreases in

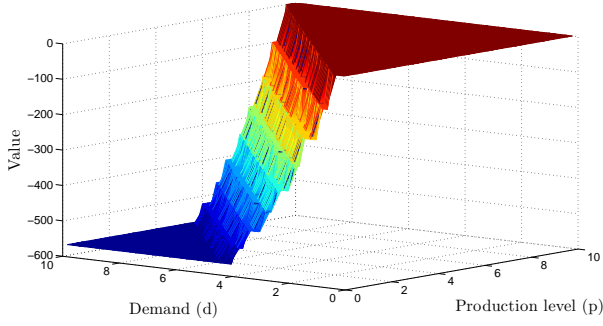


Figure 3: The optimal value function of the robust energy production game for the producer at horizon 8. The production and demand increase and decrease variables were set to $prd_u = prd_d = 1.0$ and $nat_u = nat_d = 0.5$, respectively.

a step-wise manner from the point where the production level meets demand, indicating that production levels just beneath demand have a higher value than those well below demand.

7 RELATED WORK

Solutions to stochastic games have been proposed from within both game theory and reinforcement learning. The first algorithm, game theoretic or otherwise, for finding a solution to a stochastic game was given by Shapley (Shapley, 1953). Shapley’s algorithm calculates a value function $V(s)$ over discrete states which converges to an optimal value function $V^*(s)$. $V^*(s)$ represents the expected discounted future reward if both players in the game follow the game’s Nash equilibrium. The algorithm is in essence an extension of the Value Iteration algorithm to stochastic games.

A partially implementable solution, based on reinforcement learning, for a subclass of stochastic games was first introduced by (Littman, 1994). Littman’s algorithm, Minimax-Q, extends the traditional Q-learning algorithm for MDPs to zero-sum discrete stochastic games. The algorithm converges to the stochastic game’s equilibrium solution. Hu and Wellman (Hu & Wellman, 1998) extended Minimax-Q to general-sum games and proved that it converges to a Nash equilibrium under certain restrictive conditions. Although both reinforcement learning based algorithms are able to calculate equilibrium solutions they are limited to discrete state formulations of stochastic games. In this paper we provide the first known exact closed-form solution to a subclass of continuous state zero-sum stochastic games defined by piecewise constant reward and piecewise linear transition functions.

Several techniques have been put forward to tackle contin-

uous state spaces in MDPs. Li and Littman (Li & Littman, 2005) describe a method for approximate solutions to continuous state MDPs. In their work, Li and Littman only allow for rectilinearly aligned constraints in their reward and transition functions, not arbitrary linear constraints, and cannot handle general linear transition models without approximation. Our SDP method provides exact solutions without these restrictions, which makes SDP strictly more general. Also, Li and Littman did not consider game-theoretic extensions of their work or the parameterised optimisation problem that these extensions entail.

Symbolic dynamic programming techniques have been previously used to calculate exact solutions to single agent MDPs with both continuous state and actions in a variety of non-game theoretic domains (Sanner *et al.*, 2011; Zamani & Sanner, 2012). In this paper we build on this work and present the first application of SDP to stochastic games with concurrently acting agents.

8 CONCLUSIONS

In this paper we have characterised a subclass of zero-sum continuous stochastic games that can be solved exactly via parameterised linear optimisation. We have also presented a novel symbolic dynamic programming algorithm that can be used to calculate exact solutions to this subclass of games for arbitrary horizons. The algorithm was used to calculate the first known exact solutions to a variety of continuous stochastic games with piecewise constant reward and piecewise linear transitions.

There are a number of avenues for future research. Firstly, it is important to examine more general representations of the reward and transition functions while still guaranteeing exact solutions. Another direction of research lies within improving the scalability of the algorithm by either extending techniques for Algebraic Decision Diagrams (Bahar *et al.*, 1993) from APRICODD (St-Aubin *et al.*, 2000) under the current restrictions on the reward and transition functions, bounded error compression for XADDs (Vianna *et al.*, 2013) for more expressive representations, or lazy approximation of value functions as piecewise linear XADDs (Li & Littman, 2005). Search based approaches such as RTDP (Barto *et al.*, 1995) and HAO* (Meuleau *et al.*, 2009) are also readily adaptable to SDP. These extensions may then be used to model sequential decision making in more complex financial and economic domains. Finally, SDP can be used to calculate exact solutions to general sum stochastic games. The advances made within this paper open up a number of potential novel research paths which may be used to progress solutions to sequential game theoretic domains with continuous state.

References

- Bahar, R.I., Frohm, E.A., Gaona, C.M., Hachtel, G.D., Macii, E., Pardo, A., & Somenzi, F. 1993. Algebraic Decision Diagrams and Their Applications. *Journal of Formal Methods in Systems Design*, **10**, 171–206.
- Barto, Andrew G., Bradtke, Steven J., & Singh, Satinder P. 1995. Learning to Act using Real-Time Dynamic Programming. *Artificial Intelligence*, **72**(1-2), 81–138.
- Bellman, Richard E. 1957. *Dynamic Programming*. Princeton, NJ: Princeton University Press.
- Bennett, Kristin P., & Mangasarian, O. L. 1993. Bilinear Separation of Two Sets in N-space. *Comput. Optim. Appl.*, **2**(3), 207–227.
- Bertsekas, Dimitri P. 1987. *Dynamic Programming: Deterministic and Stochastic Models*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc.
- Boutilier, Craig, Reiter, Ray, & Price, Bob. 2001. Symbolic Dynamic Programming for First-order MDPs. *Pages 690–697 of: Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01)*. IJCAI, vol. 1.
- Howard, Ronald A. 1960. *Dynamic Programming and Markov Processes*. The M.I.T. Press.
- Hu, Junling, & Wellman, Michael P. 1998. Multiagent Reinforcement Learning: Theoretical Framework and an Algorithm. *Pages 242–250 of: Proceedings of the Fifteenth International Conference on Machine Learning (ICML-98)*. ICML. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.
- Li, Lihong, & Littman, Michael L. 2005. Lazy Approximation for Solving Continuous Finite-horizon MDPs. *Pages 1175–1180 of: Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI-05)*. AAAI, vol. 3. AAAI Press.
- Littman, Michael L. 1994. Markov Games as a Framework for Multi-Agent Reinforcement Learning. *Pages 157–163 of: Proceedings of the Eleventh International Conference on Machine Learning Machine Learning (ICML-94)*. ICML. San Francisco, California, USA: Morgan Kaufmann Publishers Inc.
- Meuleau, Nicolas, Benazera, Emmanuel, Brafman, Ronen I., Hansen, Eric A., & Mausam. 2009. A Heuristic Search Approach to Planning with Continuous Resources in Stochastic Domains. *Journal of Artificial Intelligence Research*, **34**, 27–59.
- Neumann, J. 1928. Zur Theorie der Gesellschaftsspiele. *Mathematische Annalen*, **100**(1), 295–320.
- Osborne, Martin J. 2004. *An Introduction to Game Theory*. New York: Oxford University Press.
- Petrik, Marek, & Zilberstein, Shlomo. 2011. Robust Approximate Bilinear Programming for Value Function Approximation. *Journal of Machine Learning Research*, **12**, 3027–3063.
- Sanner, Scott, Delgado, Karina, & Nunes de Barros, Leliane. 2011. Symbolic Dynamic Programming for Discrete and Continuous State MDPs. *Pages 1–10 of: Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence (UAI-11)*. UAI.
- Shapley, L. S. 1953. Stochastic Games. *Proceedings of the National Academy of Sciences*, **39**(10), 1095–1100.
- St-Aubin, Robert, Hoey, Jesse, & Boutilier, Craig. 2000. APRICODD: Approximate Policy Construction Using Decision Diagrams. *Pages 1089 – 1095 of: Advances in Neural Information Processing 13 (NIPS 2000)*. NIPS. Denver, Colorado, USA: MIT Press.
- Vianna, Luis Gustavo Rocha, Sanner, Scott, & Nunes de Barros, Leliane. 2013. Bounded Approximate Symbolic Dynamic Programming for Hybrid MDPs. *Pages 1–9 of: Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence (UAI-13)*. UAI.
- Zamani, Zahra, & Sanner, Scott. 2012. Symbolic Dynamic Programming for Continuous State and Action MDPs. *Pages 1–7 of: Proceedings of the Twenty-Sixth Conference on Artificial Intelligence (AAAI-12)*. AAAI.

Recursive Best-First AND/OR Search for Optimization in Graphical Models

Akihiro Kishimoto and Radu Marinescu
IBM Research – Ireland
{akihirok, radu.marinescu}@ie.ibm.com

Abstract

The paper presents and evaluates the power of limited memory best-first search over AND/OR spaces for optimization tasks in graphical models. We propose *Recursive Best-First AND/OR Search with Overestimation* (RBFAOO), a new algorithm that explores the search space in a best-first manner while operating with restricted memory. We enhance RBFAOO with a simple overestimation technique aimed at minimizing the overhead associated with re-expanding internal nodes and prove correctness and completeness of RBFAOO. Our experiments show that RBFAOO is often superior to the current state-of-the-art approaches based on AND/OR search, especially on very hard problem instances.

1 INTRODUCTION

Graphical models provide a powerful framework for reasoning with probabilistic information. These models use graphs to capture conditional independencies between variables, allowing a concise knowledge representation and efficient graph-based query processing algorithms. Combinatorial optimization tasks such as MAP or marginal MAP inference arise in many applications and are typically tackled with either search or inference algorithms (Pearl, 1988; Dechter, 2003). The most common search scheme is the *depth-first branch and bound*. Its use for finding exact solutions was studied and evaluated extensively in the context of AND/OR search spaces that are sensitive to the underlying problem structure (Marinescu and Dechter, 2009a,b).

Meanwhile, *best-first* search algorithms, despite their better time efficiency than depth-first search (Dechter and Pearl, 1985), are largely ignored in practice primarily due to their inherently enormous memory requirements (Marinescu and Dechter, 2009b). Furthermore, an important best-first search property, avoiding the exploration of unbounded paths, seems irrelevant to optimization tasks in

graphical models where all solutions are at the same depth (ie, the number of variables).

We aim at inheriting the advantages of both depth-first and best-first search schemes. We introduce RBFAOO, a new algorithm that explores the context minimal AND/OR search graph associated with a graphical model in a *best-first* manner (even with non-monotonic heuristics) while operating within restricted memory. RBFAOO extends Recursive Best-First Search (RBFS) (Korf, 1993) to graphical models and thus uses a threshold controlling technique to drive the search in a depth-first like manner while using the available memory to cache and reuse partial search results. In addition, RBFAOO employs an overestimation method designed to further reduce the high overhead caused by re-expanding internal nodes. RBFAOO is also related to the AND/OR search algorithms based on proof/disproof numbers (Allis et al., 1994) (eg, df-pn and df-pn⁺ (Nagai, 2002)) which are very popular for solving two-player zero-sum games. However, since game solvers ignore the solution cost, they do not come with optimality guarantee. Moreover, while df-pn’s completeness on finding suboptimal solutions assumes that the cache table preserves all search results of nodes previously explored (Kishimoto and Müller, 2008), RBFAOO is proven to be complete with a small cache table. We evaluate empirically RBFAOO on benchmark problems used during the PASCAL2 Inference Challenge. Our results show that RBFAOO is often superior to the state-of-the-art solvers based on AND/OR search, especially on the hardest problem instances.

2 PRELIMINARIES

We consider combinatorial optimization problems defined over graphical models, including Bayesian networks and Markov random fields (Pearl, 1988; Koller and Friedman, 2009). A *graphical model* is a tuple $\mathcal{M} = \langle \mathbf{X}, \mathbf{D}, \mathbf{F}, \otimes \rangle$, where $\mathbf{X} = \{X_1, \dots, X_n\}$ is a set of variables and $\mathbf{D} = \{D_1, \dots, D_n\}$ is the set of their finite domains of values. $\mathbf{F} = \{f_1, \dots, f_r\}$ is a set of positive real-valued functions defined on subsets of variables, called *scopes* (ie,

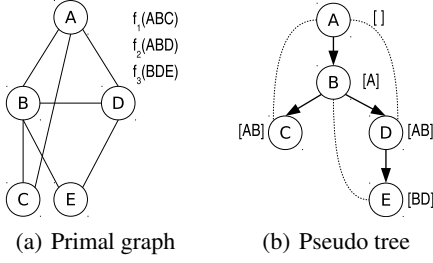


Figure 1: A simple graphical model.

$\forall j f_j : \mathbf{Y}_j \rightarrow \mathbb{R}^+$ and $\mathbf{Y}_j \subseteq \mathbf{X}$). The set of function scopes implies a *primal graph* whose vertices are the variables and which includes an edge connecting any two variables that appear in the scope of the same function. The combination operator $\otimes \in \{\prod, \sum\}$ defines the complete function represented by the graphical model \mathcal{M} as $\mathcal{C}(\mathbf{X}) = \otimes_{j=1}^r f_j(\mathbf{Y}_j)$. In this paper, we focus on *min-sum problems*, in which we would like to compute the optimal value C^* and/or its optimizing configuration x^* :

$$C^* = \mathcal{C}(x^*) = \min_{\mathbf{X}} \sum_{j=1}^r f_j(\mathbf{Y}_j) \quad (1)$$

Koller and Friedman (2009) convert the MAP task defined by $\max_{\mathbf{X}} \prod_j f_j$ to log-space and solve it as an *energy minimization* (min-sum) problem to avoid numerical issues.

2.1 AND/OR SEARCH SPACES

Dechter and Mateescu (2007) introduce the concept of AND/OR search spaces for graphical models. A *pseudo tree* of the primal graph defines the search space and captures problem decomposition.

Definition 2.1. A pseudo tree of an undirected graph $G = (V, E)$ is a directed rooted tree $\mathcal{T} = (V, E')$, such that every arc of G not included in E' is a back-arc in \mathcal{T} , namely it connects a node in \mathcal{T} to an ancestor in \mathcal{T} . The arcs in E' may not all be included in E .

Given a graphical model $\mathcal{M} = \langle \mathbf{X}, \mathbf{D}, \mathbf{F}, \sum \rangle$ with primal graph G and a pseudo tree \mathcal{T} of G , the *AND/OR search tree* $S_{\mathcal{T}}$ based on \mathcal{T} has alternating levels of OR nodes corresponding to the variables and AND nodes corresponding to the values of the OR parent's variable, with edges weighted according to \mathbf{F} . Identical subproblems, identified by their *context* (the partial instantiation that separates the subproblem from the rest of the problem graph), can be merged, yielding an *AND/OR search graph* (Dechter and Mateescu, 2007). Merging all context-mergeable nodes yields the *context minimal AND/OR search graph*, denoted by $C_{\mathcal{T}}$. The size of the context minimal AND/OR graph is exponential in the induced width of G along a depth-first traversal of \mathcal{T} (Dechter and Mateescu, 2007).

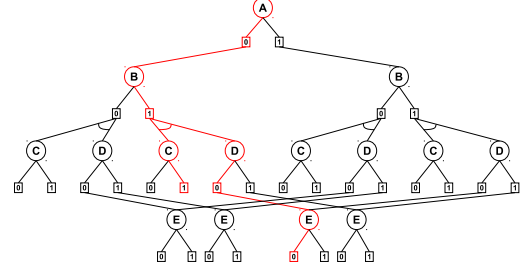


Figure 2: Context minimal AND/OR search graph.

A *solution tree* T of $C_{\mathcal{T}}$ is a subtree such that: (1) it contains the root node of $C_{\mathcal{T}}$; (2) if an internal AND node n is in T then all its children are in T ; (3) if an internal OR node n is in T then exactly one of its children is in T ; (4) every tip node in T (ie, nodes with no children) is a terminal node. The cost of a solution tree is the sum of the weights associated with its arcs.

Each node n in $C_{\mathcal{T}}$ is associated with a *value* $v(n)$ capturing the optimal solution cost of the conditioned subproblem rooted at n . It was shown that $v(n)$ can be computed recursively based on the values of n 's successors: OR nodes by minimization, AND nodes by summation (see also (Dechter and Mateescu, 2007)).

Example 1. Figure 1(a) shows a simple graphical model with 5 bi-valued variables $\{A, B, C, D, E\}$ and 3 functions $\{f_1(ABC), f_2(ABD), f_3(BDE)\}$, respectively. Figure 2 displays the context minimal AND/OR search graph based on the pseudo tree from Figure 1(b). The contexts of the variables are shown next to the corresponding pseudo tree nodes. A solution tree corresponding to the assignment $(A = 0, B = 1, C = 1, D = 0, E = 0)$ is highlighted.

2.2 AND/OR SEARCH ALGORITHMS

AND/OR Branch and Bound (AOBB) (Marinescu and Dechter, 2009a,b) is a state-of-the-art informed search approach for solving optimization tasks over graphical models. AOBB explores in a *depth-first* manner the context minimal AND/OR search graph associated with the problem and therefore takes advantage of problem decomposition. During search, AOBB keeps track of the value of the best solution found so far (an upper bound on the optimal cost) and uses this value and the heuristic function to prune away portions of the search space that are guaranteed not to contain the optimal solution in a typical branch and bound manner. Most notably, AOBB guided by a class of partitioning based heuristics won the first place in the PASCAL2 competition (Otten et al., 2012).

Best-First AND/OR Search (Marinescu and Dechter, 2009b) (AOBF) is a variant of AO* (Nilsson, 1980) applicable to graphical models that explores the graph in a *best-first* rather than depth-first manner. This enables AOBF to visit a significantly smaller search space than AOBB

which sometimes translates into important time savings. Extensive empirical evaluations (Marinescu and Dechter, 2009b) showed that when given enough memory AOBF is often superior to AOBB. However, in many practical situations AOBF’s overhead of maintaining in memory the explicated portion of the search space is still prohibitively large. AOBB therefore remains the best alternative.

3 RECURSIVE BEST-FIRST AND/OR SEARCH WITH OVERESTIMATION

We introduce RBFAOO, a new algorithm that belongs to the class of recursive best-first search algorithms and employs a local threshold controlling mechanism to explore the context minimal AND/OR search graph in a depth-first like manner (Korf, 1993; Nagai, 2002). It can however use additional memory to cache and reuse partial search results to enhance performance. RBFAOO also leverages an overestimation technique to possibly find a suboptimal solution and then refine it to an optimal one. The latter plays an essential role in enhancing the performance by avoiding a high overhead of re-expanding internal nodes.

Before explaining RBFAOO in detail, we give an overview of the threshold controlling scheme that makes RBFAOO behave similarly to AO*. Assume that the weight from an OR node to an AND node is 1, the weight from an AND node to an OR node is 0, and a heuristic function h returns values as shown in Figure 3. Let $q(n)$, called *q-value*, be a lower bound of the solution cost at node n and $th(n)$ be RBFAOO’s threshold at n . RBFAOO keeps examining the subtree rooted at n until either $q(n) > th(n)$ or the subtree is solved optimally. In Figure 3(a), RBFAOO selects B to expand, because $w(A, B) + q(B) = w(A, B) + h(B) = 3 < w(A, C) + q(C) = w(A, C) + h(C) = 5$. It sets $th(B) = w(A, C) + q(C) - w(A, B) = 4$ to indicate that C becomes the best child (ie, $w(A, B) + q(B) > w(A, C) + q(C)$ holds) if $q(B) > th(B)$. Then, RBFAOO expands B and updates $q(B)$ by using the q -values of B ’s children (Figure 3(b)). Because $q(B) = q(D) + q(E) = h(D) + h(E) = 3$, $q(B) \leq th(B)$ still holds. Hence, RBFAOO examines B ’s descendants with no backtracks to A . Assume that D is chosen to examine. RBFAOO sets $th(D) = th(B) - q(E) = 2$ to indicate that C becomes best if $q(D) > th(D)$ holds, which is equivalent to $q(B) > th(B)$, because $q(B) = q(D) + q(E)$ and $th(B) = th(D) + q(E)$. Next, RBFAOO expands D and updates $q(D) = w(D, F) + h(F) = 4$ (Figure 3(c)). Because $q(D) > th(D)$, the subtree rooted at D contains no best leaf in terms of AO*’s strategy. RBFAOO backtracks to A by updating $q(B) = q(D) + q(E) = 6$ and examines C (Figure 3(d)) with $th(C) = w(A, B) + q(B) - w(A, C) = 6$ to be able to select B when B becomes best.

RBFAOO gradually grows its search space by updating the q -values of internal nodes and re-expanding them. The

overhead of internal node re-expansions is still high, even if RBFAOO does not always propagate back the q -values. For example, assume that an internal OR node n has two children c_1 and c_2 , c_1 is selected to re-expand, and RBFAOO proves that c_2 becomes best to examine after expanding only one leaf that is k -steps away from c_1 . If k is large, RBFAOO need to spend most of time in re-expanding internal nodes without exploring the new search space. The overestimation technique avoids this scenario by increasing the threshold while it verifies solution optimality.

3.1 ALGORITHM DESCRIPTION

Figure 4 shows the pseudo-code of RBFAOO. Let ϵ be a small number and assume $\infty - \epsilon < \infty$. In practice, a finite real number is used to represent ∞ . Let δ be an empirically tuned parameter that determines the amount of overestimation. *HasNoChildren* checks whether a node has no children (ie, terminal leaf or dead-end) or not. *Evaluate* evaluates a terminal leaf/dead-end n and returns a pair of the cost (ie, 0 or ∞) and a Boolean flag indicating whether n is solved or a dead-end. *UnsolvedChild* returns an unsolved child. *SaveInCache* saves in the cache table a q -value and a flag indicating whether a node is solved optimally or not. *RetrieveFromCache* retrieves them from the cache table. *Context* calculates the context of a node.

When RBFAOO starts solving a problem, the threshold of the root node is set to $\infty - \epsilon$. If RBFAOO exceeds this threshold, the problem is proven to have no solution. Otherwise, RBFAOO returns the optimal solution cost to the problem. In addition, RBFAOO can be easily instrumented to recover the assignment corresponding to the optimal solution cost (this extension is omitted for clarity reasons).

Function *RBFS*(n) traverses the subtree rooted at n in a depth-first manner. It calculates either an optimal solution cost or a lower bound by using *BestChild* or *Sum* and checks if the termination condition is satisfied. If the solution optimality is guaranteed at n , $n.solved$ is set to **true**.

At an OR node, *RBFS*(n) may find a suboptimal solution. In this case, $n.solved$ is still set to **false** and *RBFS*(n) continues examining other children until it finds an optimal solution at n . Because the solution cost found so far is an upper bound of the optimal one, *RBFS*(n) uses that solution cost (maintained by *ub*) to prune away unpromising branches and to adjust the threshold.

When *RBFS*(n) selects c_{best} , it examines c_{best} with a new threshold. At OR nodes, $c_{best}.th$ is set to subtracting the weight between n and c_{best} from the minimum of:

1. The current threshold for n .
2. The second smallest lower bound q_2 to solve n ’s child with considering the weight from n to that child among a list of such lower bounds of n ’s children.

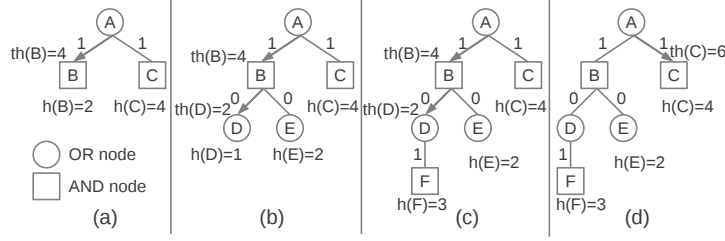


Figure 3: Snapshot of RBFAOO without overestimation.

This indicates when the current second best child becomes the best one. Additionally, a parameter δ called *overestimation rate*, that allows for returning a suboptimal solution cost is added to q_2 to avoid an excessive number of backtracks to n .

3. The upper bound of the optimal solution at n .

At AND nodes, $c_{best}.th$ is set to the sum of c_{best} 's q-value and the gap between $n.th$ and the total q-value of n 's children. If $q(c_{best}) > c_{best}.th$, $q(n) > n.th$ also holds.

Let N be the number of nodes in the search space. If the search space fits into memory, AO* expands $O(N)$ nodes in the worst case. In contrast, due to node re-expansions, RBFAOO's worst-case scenario is $O(N^2)$. However, in practice, by introducing δ , RBFAOO avoids such a high node re-expansion overhead (see Section 4).

3.2 IMPLEMENTATION DETAILS

The cache table is implemented as a hash table with the Zobrist function (Zobrist, 1970) using 96-bit integers. The Zobrist function computes almost uniformly distributed hash keys by XORing precomputed random integers, each of which represents the component of a context and is commonly used in game-playing programs and in planning. Each cache table entry preserves the context of a node to avoid collisions caused by an astronomically small possibility of two different nodes having the same hash key.

When RBFAOO fills up the cache table and tries to store new results there, some cached results must be replaced. We use SmallTreeGC (Nagai, 1999), a batch-based replacement that discards $R\%$ of the table entries with small subtree sizes. We set R to 30.

Due to floating-point errors, RBFAOO is occasionally unable to expand a new leaf. We bypass this by using small error margins when comparing floating-point numbers.

As in full RBFS (Korf, 1993), q-values of children can be increased based on the current q-value of their parent. This technique generates more accurate heuristic information when cache table entries are replaced or non-monotonic heuristic functions are used. We implemented this tech-

nique and observed small performance improvement.

3.3 CORRECTNESS AND COMPLETENESS

We prove that RBFAOO is both correct and complete for solving optimization tasks defined over graphical models even with a small cache table and arbitrary cache table replacement schemes as long as certain table entries are preserved. In contrast, $df-pn^+$ may return suboptimal solutions if the contexts of nodes are used. Additionally, although its completeness on finding either one (possibly suboptimal) or no solution is proven, $df-pn^+$ must preserve all TT entries and the completeness with TT replacement schemes remains an open question (Kishimoto and Müller, 2008). The following theorems hold with/without enhancements described in Section 3.2.

Theorem 3.1 (correctness). *Given a graphical model $\mathcal{M} = \langle \mathbf{X}, \mathbf{D}, \mathbf{F}, \sum \rangle$, if RBFAOO solves \mathcal{M} with admissible heuristic function h , its solution is always optimal.*

Proof. Let $v(n)$ be the optimal solution cost for node n . We first prove that for any value q for node n in the cache table, $q \leq v(n)$ holds. Since different nodes with the same context are proven to be equivalent in DAGs (Marinescu and Dechter, 2009b), we denote n as $\text{Context}(n)$ when a search result at node n is saved in the cache table. Additionally, we assume $h(n) = h(n')$ if $\text{Context}(n) = \text{Context}(n')$.

Let Cache_t be the state of the cache table immediately after the t -th save is performed in the cache table. Let $Q_t(n)$ be the value saved in Cache_t for n if that value exists in Cache_t or $h(n)$ if n is not preserved in Cache_t . By induction on t , we prove that all the entries in cache table contain values that do not overestimate optimal ones.

1. Because no result is stored in Cache_0 , the above property holds for $t = 0$.
2. Assume that the above property holds for $t = k$. $Q_{k+1}(n)$, saved in Cache_{k+1} , is then calculated as:
 - If n is a terminal leaf, $\text{Evaluate}(n)$ in the pseudo code always returns $v(n)$. $Q_{k+1}(n) = v(n)$ therefore holds.

```

// Set up for the root node
double RBFAOO(node root) {
    root.th =  $\infty - \epsilon$ ;
    q = RBFS(root);
    return q;
}
// Depth-first search with a threshold
double RBFS(node n) {
    // Terminal leaf/dead-end check
    if (HasNoChildren(n)) {
        // Calculate the probability
        // for a terminal leaf or dead-end
        (q, s) = Evaluate(n);
        // Store search results
        SaveInCache(Context(n), q, s);
        return q;
    }
    GenerateChildren(n);
    // Continue search until satisfying
    // the termination condition
    if (n is an OR node)
        loop {
            (cbest, q, q2, ub) = BestChild(n);
            if (n.th < q || n.solved = true)
                break;
            // Update the threshold
            cbest.th = min(n.th,
                          q2 +  $\delta$ ,
                          ub) - w(n, cbest);
            RBFS(cbest);
        }
    else
        loop { // AND node
            q = Sum(n);
            if (n.th < q || n.solved = true)
                break;
            (cbest, qcbest) = UnsolvedChild(n);
            // Update the threshold
            cbest.th = n.th - (q - qcbest);
            RBFS(cbest);
        }
    // Store search results
    SaveInCache(Context(n), q, n.solved);
    return q;
}

// Select the best child
double BestChild(node n) {
    q = q2 = ub =  $\infty$ ;
    n.solved = false;
    foreach (n's child ci) {
        ct = Context(ci);
        if (ct is in the cache table)
            (qci, s) = RetrieveFromCache(ct);
        else {
            qci = h(ci);
            s = false;
        }
        qci = w(n, ci) + qci;
        if (s = true) // ci is solved
            ub = min(ub, qci);
        if (qci < q ||
            (qci = q && n.solved = false)) {
            q2 = q;
            n.solved = s;
            q = qci;
            cbest = ci;
        } else if (qci < q2)
            q2 = qci;
    }
    return (cbest, q, q2, ub);
}
// Calculate the total value
double Sum(node n) {
    q = 0;
    n.solved = true;
    foreach (n's child ci) {
        ct = Context(ci);
        if (ct is in the cache table)
            (qci, s) = RetrieveFromCache(ct);
        else {
            qci = h(ci);
            s = false;
        }
        q = q + qci;
        n.solved = n.solved  $\wedge$  s;
    }
    return q;
}

```

Figure 4: Pseudo-code of RBFAOO

- If n is an internal OR node, $Q_{k+1}(n) = w(n, c_{best}) + Q_k(c_{best}) = \min_i (w(n, c_i) + Q_k(c_i))$ holds where c_i is n 's child. Additionally, because $Q_k(c_i) \leq v(c_i)$, $Q_{k+1}(n) \leq \min_i (w(n, c_i) + v(c_i)) = v(n)$ holds.
- If n is an internal AND node, $Q_{k+1}(n) = \sum_i Q_k(c_i)$ where c_i is n 's child. Since $Q_k(c_i) \leq v(c_i)$, $Q_{k+1}(n) \leq \sum_i v(c_i) = v(n)$ holds.

Hence, $Q_t(n) \leq v(n)$ holds in case of $t = k + 1$.

Let $Q(\text{root})$ be a value that is about to be saved in the cache table with satisfying the termination condition of

$\text{root.solved} = \text{true}$. $Q(\text{root}) \leq v(\text{root})$ holds from the above. Additionally, because RBFAOO has traced a solution tree with the cost of $Q(\text{root})$, $v(\text{root}) \leq Q(\text{root})$ holds. Therefore, $Q(\text{root}) = v(\text{root})$ holds. \square

Theorem 3.2 (completeness). *Let $\mathcal{M} = \langle \mathbf{X}, \mathbf{D}, \mathbf{F}, \Sigma \rangle$ be a graphical model with primal graph G , let \mathcal{T} be a pseudo tree G and let $C_{\mathcal{T}}$ be the context minimal AND/OR search graph based on \mathcal{T} (also a finite DAG). Assume that RBFAOO preserves the q -values of the nodes n_1, n_2, \dots, n_k which are on the current search path and the q -values of n_i 's siblings. Then RBFAOO eventually returns an optimal solution or proves no solution exists.*

Proof sketch. Let a *marked* node be a node expanded at least once by RBFAOO and an *unmarked* node be a node that has never been expanded. Denote $p_1 \subset p_2$ if the path length of p_2 is longer than that of p_1 and p_2 is identical to p_1 if p_2 is limited to the path with the length of p_1 . Denote $p_1 \not\subset p_2$ unless it holds $p_1 \subset p_2$. Let $s(n, p)$ be the sum of the edge costs from the root to n via path p . Assume that $s(n, p) + q(n) < \infty - \epsilon$ holds for any path p if $q(n) \neq \infty$, which is reasonable in practical settings.

Let $q_{t,p}(n)$ and $th_{t,p}(n)$ be the q -value and threshold for t -th visit to n via path p , respectively. Assume that RBFAOO expands no unmarked nodes after expanding k unmarked nodes. Then, because the search space is finite, there are two unproven marked nodes n and m examined as follows:

1. RBFAOO starts searching downward from n via path p_1 since $q_{t_1,p_1}(n) \leq th_{t_1,p_1}(n)$ holds.
2. RBFAOO reaches m via path p_2 that satisfies $th_{t_2,p_2}(m) < q_{t_2,p_2}(m)$ and $p_1 \subset p_2$.
3. RBFAOO keeps exploring the remaining search space rooted at n via p_1 (and composed of marked nodes) and backtracks to n .
4. Continue steps (1)-(3).

If n is an AND node, satisfying $th_{t',p_3}(c) < q_{t',p_3}(c)$ immediately leads to satisfying $th_{t,p_1}(n) < q_{t,p_1}(n)$ where c is n 's unproven child and $p_1 \subset p_3$. Because backtracking to n 's parent contradicts step (1), n is an OR node. Additionally, $th_{t',p_3}(c) < th_{u,p_3}(c)$ holds for any unproven child c , $t' < u$ and $p_1 \subset p_3$, because the q -values of n 's children are preserved in memory as described in the assumption of the theorem. With similar discussions, there is an infinite sequence $u_1, u_2, \dots, u_k, \dots$ that satisfies $u_i < u_j$ for $i < j$, $th_{u_i,p_2}(m) < th_{u_j,p_2}(m)$ and $th_{u_i,p_2}(m) < q_{u_i,p_2}(m)$. This indicates that m has at least one unproven child o_1 via path r_1 ($p_2 \not\subset r_1$) that contributes to increasing $q_{u_i,p_2}(m)$ and satisfying $th_{u_i,p_2} < q_{u_i,p_2}(m)$ when $q_{u_i,p_2}(m)$ is calculated. Because the search space is DAG, $q(o_1)$ is never affected by $q(m)$. With similar discussions, if no unmarked node is expanded, there is an infinite number of nodes $o_1, o_2, \dots, o_k, \dots$, where o_{j+1} is a child of o_j that contributes to increasing $q(o_j)$ and satisfying $th(o_j) < q(o_j)$. However, this contradicts the assumption of the finite search space. Hence, by eventually examining the whole search space, RBFAOO finds an optimal solution (see Theorem 3.1) or proves no solution. \square

4 EXPERIMENTS

We empirically evaluate our proposed best-first search scheme on the MAP task in graphical models. We compare

RBFAOO against the state-of-the-art depth-first and best-first AND/OR search solvers proposed recently in (Marinescu and Dechter, 2009b) and denoted by AOBB and AOBF, respectively. All competing algorithms use pre-compiled mini-bucket heuristics (Kask and Dechter, 2001; Marinescu and Dechter, 2009b) for guidance and are restricted to a static variable ordering obtained as a depth-first traversal of a minfill pseudo tree (Marinescu and Dechter, 2009a). Since AOBF cannot use an initial upper bound (obtained via local search) we also disabled its use by AOBB and RBFAOO in order to maintain a fair comparison.

Our benchmark problems¹ include three sets of instances from genetic linkage analysis (Fishelson and Geiger, 2002) (denoted pedigree), grid networks and protein side-chain interaction networks (denoted pdb) (Yanover et al., 2008). In total, we evaluated 21 pedigrees, 32 grids and 240 protein networks. The algorithms were implemented in C++ (64-bit) and the experiments were run on a 2.6GHz 8-core processor with 80GB of RAM.

We report the CPU time in seconds and the number of nodes expanded for solving the problems. We also specify the problems parameters such as the number of variables (n), maximum domain size (k), the depth of the pseudo tree (h) and the induced width of the graph (w^*). The best performance points are highlighted. In each table, 'oom' stands for out-of-memory and '-' denotes out-of-time. Note that oom for RBFAOO/AOBB indicates that the mini-bucket heuristic pre-computation procedure uses up the physical memory before search is performed.

Tables 1 and 2 show the results obtained for experiments with pedigree, grid and protein networks. For space reasons and clarity we select a representative subset from the full 293 instances. The columns are indexed by the mini-bucket i -bound which ranged between 6 and 16 for pedigrees and grids, and between 2 and 5 for proteins, respectively. All algorithms were allotted a 1 hour time limit. Algorithm AOBF(i) was allowed a maximum of 80GB of RAM while algorithm RBFAOO(i) used a 10-20GB cache table with 134,217,728 entries pre-allocated before search. The overestimation parameter δ was set to 1.

We observe clearly that RBFAOO(i) improves considerably over its competitors, especially at relatively small i -bounds which yield relatively weak heuristics. For example, on the pedigree30 instance, RBFAOO(6) with the smallest reported i -bound ($i = 6$) was 4 and 41 times faster than AOBF(6) and AOBB(6), respectively. Similarly, RBFAOO(6) solves the 75-23-5 grid in about 30 minutes and expands over 300 million nodes, while both AOBB(6) and AOBF(6) run out of time and memory, respectively. As the i -bound increases and the heuristics become more accurate thus pruning the search space more effectively, the differences in running time between the algorithms de-

¹ All instances are available at <http://graphmod.ics.uci.edu>.

Table 1: CPU time (seconds) and number of nodes expanded for pedigree and grid networks. Time limit 1 hour. RBFAOO(i) ran with a 10-20GB cache table (134,217,728 entries) and overestimation parameter $\delta = 1$.

instance (<i>n, k, w*, h</i>)	algorithm	<i>i</i> = 6		<i>i</i> = 8		<i>i</i> = 10		<i>i</i> = 12		<i>i</i> = 14		<i>i</i> = 16	
		time	nodes	time	nodes	time	nodes	time	nodes	time	nodes	time	nodes
pedigree instances													
pedigree7 (1068,4,28,140)	AOBB		-		-		-		-		-		-
	AOBF		-		-		-		-		-		-
	RBFAOO		oom		oom		oom	2210	345204317	1368	216767091	818	144733023
pedigree9 (1118,7,25,123)	AOBB		-		-		-		-		-	1076	139749607
	AOBF		oom		oom	1846	30506650	1379	20960401	1152	20897564	263	7682927
	RBFAOO	1084	195214857	728	136764248	522	97410715	248	46922921	241	44561263	60	10634230
pedigree13 (1077,3,30,125)	AOBB		-		-		-		-		-		-
	AOBF		oom		oom		oom		oom		oom		oom
	RBFAOO		-		-		-		-		-	2629	364037130
pedigree19 (793,5,21,51)	AOBB		-		-		-		-		-		-
	AOBF		oom		oom		oom		oom		oom		oom
	RBFAOO		-		-	1753	319268527	834	168262596	226	45738797	378	69780223
pedigree30 (1289,5,20,105)	AOBB		-		-		-		-		-		-
	AOBF	825	113195179	1450	198371250	244	34182326	63	10855277	102	17794376	3	107437
	RBFAOO	83	2648120	103	2689106	45	1717523	24	867988	39	932986	3	30794
pedigree39 (1272,5,20,77)	AOBB		-		-		-		-		-		-
	AOBF	20	5435997	19	5401921	14	3840692	5	1406493	6	1691396	3	60479
	RBFAOO		-	935	125740961	107	15616376	5	885551	9	1272810	5	24174
pedigree41 (1062,5,29,119)	AOBF	307	9740964	215	8073776	53	2347928	8	384757	14	607860	5	19960
	RBFAOO	79	19804239	67	16260143	14	3461943	2	480866	4	666873	5	24826
	RBFAOO		oom		oom		oom		oom		oom		oom
binary grid instances													
50-20-5 (400,2,27,97)	AOBB		-		-		-		-		-		-
	AOBF		oom		oom		oom		oom	1309	33138951	789	19857843
	RBFAOO	1163	214829892	736	142564959	385	80803927	232	48848448	120	25641963	66	13994679
75-20-5 (400,2,27,99)	AOBB		-	738	111785572	309	43858649	36	5997367	19	3234878	10	1405451
	AOBF	2268	30767273	567	20761132	182	5685498	42	1766240	23	930839	12	442278
	RBFAOO	212	46289779	89	19603768	39	8451032	9	2026625	5	1007726	4	511806
75-22-5 (484,2,30,107)	AOBB		-		-		-	2206	314621887	994	144092486	67	10500198
	AOBF		oom		oom	1123	34528523	743	22103512	313	10577016	49	1714348
	RBFAOO	563	107126385	643	118981360	227	44947693	153	29325424	91	18077594	17	3047665
75-23-5 (529,2,31,122)	AOBB		-		-		-		-		-	131	16039678
	AOBF		oom		oom	1751	39532238	417	11103193	340	8092564	37	1218023
	RBFAOO	1860	304935340	1109	198613807	455	87285533	106	20952230	71	13910863	17	2915543
75-26-5 (676,2,36,134)	AOBB		-		-		-		-		-		-
	AOBF		oom		oom		oom		oom		oom		oom
	RBFAOO		-		-		-		-	3005	394135020		oom
90-23-5 (529,2,31,116)	AOBB		-		-	1479	195188949	560	74507590	51	7366618	102	13921196
	AOBF	970	32478634	376	12937697	289	10087022	91	3169720	52	1944481	33	1224632
	RBFAOO	277	52920346	105	20736738	71	14460847	18	3602104	9	1810658	9	1390189
90-26-5 (676,2,36,136)	AOBB		-		-		-		-		-	1647	186283089
	AOBF	1016	25948278	1108	29700313	505	16035732	552	16728882	457	12983459	159	4413795
	RBFAOO	241	44068170	183	33284922	68	12738955	65	12176988	49	9170451	30	5180019

crease. In terms of the size of the search spaces explored, we see that AOB(i) typically expands the smallest number of nodes, as expected. RBFAOO(i) expands more nodes than AOB(i), due to re-expansions, but in many cases it expands significantly fewer nodes than AOB(i) which translates into important time savings. We also notice that RBFAOO(i) and AOB(i) have a relatively small overhead per node expansion. On the other hand, the computational overhead of AOB(i) is much larger. It is caused primarily by maintaining an extremely large search space in memory and, secondly, because the node values are typically updated all the way up to the root. Most notably, RBFAOO(i) was the only algorithm that could solve the most difficult instances in these benchmarks (eg, pedigrees 7, 13, 19 and 41, as well as grid 75-26-5). This demonstrates the benefit of expanding nodes in best-first rather than depth-first manner as well as using efficiently a bounded amount of memory, thus overcoming the most critical limitation of AOB(i). Finally, the results on the protein networks show a similar pattern, namely RBFAOO(i) improves considerably over both AOB(i) and AOB(i) for relatively small i -bounds. This is important because, unlike the pedigrees and grids, these problems have very large domains (81 val-

ues) and therefore the mini-bucket heuristics could only be compiled for small i -bounds. Figure 5 which plots the normalized total CPU time as a function of the i -bound summarizes the running time profile of the competing algorithms across the benchmarks we considered.

In Table 3 we report on five additional very difficult genetic linkage analysis networks. The mini-bucket i -bound was set to 20 in this case. We see again that RBFAOO is the best performing algorithm closing all instances within the 100 hour time limit. In contrast, AOB could solve only one instance while AOB ran out of memory. For example, on the type4-120-17 instance, RBFAOO was nearly 3 orders of magnitude faster than AOB, while expanding 3 orders of magnitude fewer nodes.

We summarize next the most important additional factors that could help improve RBFAOO(i)’s performance.

Impact of caching: Table 4 shows the average performance of algorithm RBFAOO(i) (as CPU time in seconds, number of nodes expanded, and number of problem instances solved) as a function of available memory, across all three benchmarks. The columns are indexed by the cache table size used, namely *very small* (10-20MB), *small*

Table 2: CPU time (seconds) and number of nodes expanded for protein networks. Time limit 1 hour. RBFAOO(i) ran with a 10-20GB cache table (134,217,728 entries) and overestimation parameter $\delta = 1$.

instance (n, k, w^*, h)	algorithm	$i = 2$		$i = 3$		$i = 4$		$i = 5$	
		time	nodes	time	nodes	time	nodes	time	nodes
pdb1a3c (144,81,15,32)	AOBB	-	-	-	-	2218	65175805	-	oom
	AOBF	-	oom	-	oom	-	oom	-	oom
	RBFAOO	1915	45513907	-	-	344	259261	-	oom
pdb1aac (85,81,11,21)	AOBB	129	2919570	8	2694	204	1302	-	oom
	AOBF	2851	3195539	11	6264	205	3072	-	oom
	RBFAOO	51	1148212	8	1492	204	783	-	oom
pdb1acf (90,81,9,22)	AOBB	996	55994055	2672	162495198	16	1593	136	4767
	AOBF	-	oom	-	oom	17	4021	139	10464
	RBFAOO	22	987416	56	2553896	16	1090	137	3212
pdb1ad2 (177,81,9,33)	AOBB	259	7770890	134	3806154	657	3312980	2552	274955
	AOBF	831	1250161	394	715399	1109	1049637	2595	135265
	RBFAOO	36	1227741	42	858899	585	1218780	2543	113780
pdb1ail (62,81,8,23)	AOBB	4	177150	31	75051	610	1474224	-	oom
	AOBF	80	66207	47	15817	1728	928375	-	oom
	RBFAOO	2	78677	30	16427	599	1311325	-	oom
pdb1atg (175,81,12,39)	AOBB	-	-	6	154434	260	9348036	236	24412
	AOBF	-	oom	38	119195	632	1196429	247	30072
	RBFAOO	620	24347033	4	71446	32	430747	236	11545

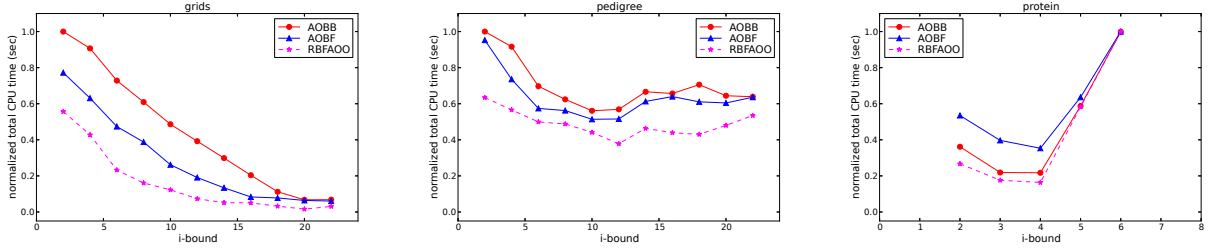


Figure 5: Normalized total CPU time as a function of the i -bound.

Table 3: CPU time (seconds) and node expansions for previously unsolved linkage networks. Time limit 100 hours.

instance	algorithm	time	nodes
type4b-100-19 (7308,5,29,354)	AOBB	-	-
	AOBF	-	oom
	RBFAOO	107258	15157422871
type4b-120-17 (7766,5,24,319)	AOBB	162196	5473951156
	AOBF	-	oom
	RBFAOO	218	3388901
type4b-130-21 (8883,5,29,416)	AOBB	-	-
	AOBF	-	oom
	RBFAOO	312887	43341893185
type4b-140-19 (9274,5,30,366)	AOBB	-	-
	AOBF	-	oom
	RBFAOO	270856	28653407450
largeFam3-10-52 (1905,3,36,80)	AOBB	-	-
	AOBF	-	oom
	RBFAOO	129633	12826083707

(100-200MB), *medium* (1-2GB) and *large* (10-20GB), respectively. We see that, as expected, as more memory is available, the performance improves considerably, namely more problem instances are solved while the running time and size of the search space decrease significantly. The best results were obtained with the 10-20GB cache.

Impact of overestimation: Figure 6 plots the CPU time, node re-expansion rate (as the ratio of the number of nodes re-expanded to the total number of expansions) and percentage of problem instances solved by RBFAOO(i) as a function of the overestimation rate δ , across all bench-

marks. We see that RBFAOO(i) without overestimation (ie, $\delta = 0$) performed rather poorly and was outperformed considerably by AOBB(i) and AOBF(i), respectively. This was due to a relatively large number of node re-expansions. On grids, for example, more than 78% of the nodes were actually re-expanded for $\delta = 0$ compared to only 11% re-expansions for $\delta = 1.2$. However, as δ increases the re-expansion rate decreases but the CPU time starts to increase due to explorations of unpromising search spaces. Therefore, we obtained the overall best performance for $\delta = 1$.

Impact of heuristics quality: Based on our empirical evaluation we noticed that RBFAOO(i) was superior to its competitors especially for relatively inaccurate heuristics (which are typically obtained for smaller i -bounds) and on the hardest problem instances. This is important because it is likely that for these types of problems it may only be possible to compute rather weak heuristics given limited resources (eg, *type4* instances in Table 3).

5 RELATED WORK

Algorithms based on proof and disproof numbers (Allis et al., 1994) have been dominating AND/OR search techniques and successfully applied to many game domains (eg, (Nagai, 2002; Kishimoto and Müller, 2005; Schaeffer et al., 2007)). See (Kishimoto et al., 2012) for a comprehensive literature review.

Table 4: Average CPU time (seconds), number of nodes expanded and number of problem instances solved by RBFAOO(i) with different cache sizes. Time limit 1 hour. $i = 10$ for grids and pedigrees, $i = 4$ for protein networks.

benchmark	10-20MB			100-200MB			1-2GB			10-20GB		
	time	nodes	solved	time	nodes	solved	time	nodes	solved	time	nodes	solved
grids	1928	496571526	15/32	1685	321943731	18/32	1257	203898268	22/32	1220	173080755	22/32
pedigree	1416	366052904	12/21	1350	277666996	13/21	1180	223258293	14/21	1155	188200810	14/21
protein	574	35439167	208/240	509	27532421	213/240	443	23135969	217/240	396	20521511	222/240

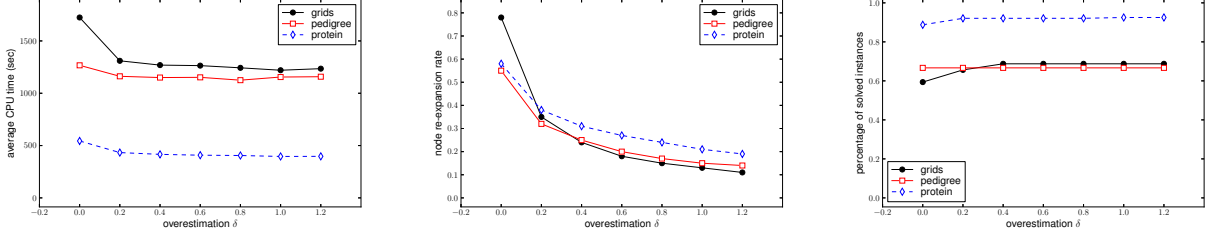


Figure 6: Average CPU time (seconds), node re-expansion rate and percentage of instances solved by RBFAOO(i) as a function of the overestimation rate δ . Time limit 1 hour. $i = 10$ for grids and pedigrees, $i = 4$ for protein networks.

A proof/disproof number estimates the difficulty of proving that the first/second player wins in a partially built tree. The proof number of node n is defined as the minimum number of leaf nodes that must be expanded to prove that the first player wins at n . A node with a smaller proof number is assumed to be easier to prove a win for the first player. In contrast, the disproof number of n is the minimum number of leaf nodes that must be expanded to prove that the second player wins at n . A node with a smaller disproof number is assumed to be easier to prove a win for the second player.

Depth-First Proof-Number Search (df-pn) (Nagai, 2002) is a depth-first reformulation of Best-First Proof-Number Search (PNS) (Allis et al., 1994) enhanced with a so-called transposition table (TT), a cache table preserving the search effort for the expanded nodes. While preserving PNS’ leaf selection strategy, df-pn empirically re-expands fewer internal nodes than PNS that always restarts from the root the procedure of finding a promising leaf to expand. Besides, df-pn runs using a small amount of space limited by the TT size in practice, although whether df-pn is complete or not with a limited amount of TT still remains an open question. As in RBFS (Korf, 1993), df-pn introduces thresholds to limit the search depth of depth-first search. Df-pn updates the thresholds of a node by taking into account when the search tree rooted at that node contains none of the most promising leaf nodes chosen by PNS’ best-first strategy. The df-pn⁺ algorithm (Nagai, 2002) generalizes df-pn by introducing evaluation functions to heuristically initialize proof and disproof numbers and a weight in each edge to decrease the overhead of node re-expansions (Kishimoto and Müller, 2005).

Other related work includes MAO* (Chakrabati et al., 1989), memory-limited AO*. Although MAO* can run under a similar memory limit to RBFAOO, it needs a spe-

cific strategy to discard examined nodes from memory. In contrast, RBFAOO can leverage arbitrary TT replacement strategies including SmallTreeGC (Nagai, 1999), which is empirically most effective in solving games. Additionally, by incorporating ideas behind RBFS and the overestimation technique, RBFAOO has much smaller overhead to update node values than MAO* and AO*.

6 CONCLUSION

The paper presents RBFAOO, a limited memory best-first AND/OR search algorithm for solving combinatorial optimization defined over graphical models. RBFAOO belongs to the Recursive Best-First Search family of algorithms and therefore uses a threshold controlling mechanism to guide the search in a depth-first like manner. It also employs a flexible caching scheme to reuse partial search results as well as an overestimation mechanism to further reduce the internal node re-expansions. We prove correctness and completeness of the algorithm. We evaluate RBFAOO empirically on a variety of benchmarks used during the PASCAL2 Inference Challenge. Our results show that RBFAOO is often superior to current state-of-the-art solvers based on AND/OR search, especially on the most difficult problem instances.

For future work we plan to extend RBFAOO to use dynamic variable orderings, an initial upper bound obtained via local search and soft arc-consistency based heuristics. One possibility we are currently investigating is to implement RBFAOO on top of the toulbar solver (de Givry et al., 2005). Since many interesting real-world problems are still too hard to solve exactly, we also plan to convert the algorithm into an anytime best-first search scheme.

REFERENCES

- L. V. Allis, M. van der Meulen, and H. J. van den Herik. Proof-number search. *Artificial Intelligence*, 66(1):91–124, 1994.
- P. Chakrabati, S. Ghose, A. Acharya, and S. de Sarkar. Heuristic search in restricted memory. *Artificial Intelligence*, 3(41):197–221, 1989.
- S. de Givry, F. Heras, J. Larrosa, and M. Zytnicki. Existential arc consistency: getting closer to full arc consistency in weighted csp. In *International Joint Conference in Artificial Intelligence (IJCAI)*, 2005.
- R. Dechter. *Constraint Processing*. Morgan Kaufmann Publishers, 2003.
- R. Dechter and R. Mateescu. AND/OR search spaces for graphical models. *Artificial Intelligence*, 171(2-3):73–106, 2007.
- R. Dechter and J. Pearl. Generalized best-first search strategies and the optimality of A*. In *Journal of ACM*, 32(3):505–536, 1985.
- M. Fishelson and D. Geiger. Exact genetic linkage computations for general pedigrees. *Bioinformatics*, 18(1):189–198, 2002.
- K. Kask and R. Dechter. A general scheme for automatic generation of search heuristics from specification dependencies. *Artificial Intelligence*, 129(1-2):91–131, 2001.
- A. Kishimoto and M. Müller. Search versus knowledge for solving life and death problems in Go. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 1374–1379, 2005.
- A. Kishimoto and M. Müller. About the completeness of depth-first proof-number search. In *Computers and Games 2008*, volume 5131 of *Lecture Notes in Computer Science*, pages 146–156, 2008.
- A. Kishimoto, M. Winands, M. Müller, and J.-T. Saito. Game-tree search using proof numbers: The first twenty years. *ICGA Journal, Vol. 35, No. 3*, 35(3):131–156, 2012.
- D. Koller and N. Friedman. *Probabilistic Graphical Models*. MIT Press, 2009.
- R. Korf. Linear-space best-first search. *Artificial Intelligence*, 62(1):41–78, 1993.
- R. Marinescu and R. Dechter. AND/OR branch-and-bound search for combinatorial optimization in graphical models. *Artificial Intelligence*, 173(16-17):1457–1491, 2009a.
- R. Marinescu and R. Dechter. Memory intensive AND/OR search for combinatorial optimization in graphical models. *Artificial Intelligence*, 173(16-17):1492–1524, 2009b.
- A. Nagai. A new depth-first search algorithm for AND/OR trees. Master’s thesis, Department of Information Science, University of Tokyo, 1999.
- A. Nagai. *Df-pn Algorithm for Searching AND/OR Trees and Its Applications*. PhD thesis, The University of Tokyo, 2002.
- N. J. Nilsson. *Principles of Artificial Intelligence*. Tioga, 1980.
- L. Otten, A. Ihler, K. Kask, and R. Dechter. Winning the PASCAL 2011 MAP challenge with enhanced and/or branch-and-bound. Technical report, School of Information and Computer Science, University of California, Irvine, 2012.
- J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.
- J. Schaeffer, N. Burch, Y. Björnsson, A. Kishimoto, M. Müller, R. Lake, P. Lu, and S. Sutphen. Checkers is solved. *Science*, 317(5844):1518–1522, 2007.
- C. Yanover, O. Schueler-Furman, and Y. Weiss. Minimizing and learning energy functions for side-chain prediction. *Journal of Computational Biology*, 15(7):899–911, 2008.
- A. L. Zobrist. A new hashing method with applications for game playing. Technical report, Department of Computer Science, University of Wisconsin, Madison, 1970.

Saturated Conditional Independence with Fixed and Undetermined Sets of Incomplete Random Variables

Henning Koehler

School of Engineering & Advanced Technology
Massey University, New Zealand
h.koehler@massey.ac.nz

Sebastian Link

Department of Computer Science
The University of Auckland, New Zealand
s.link@auckland.ac.nz

Abstract

The implication problem for saturated conditional independence statements is studied in the presence of fixed and undetermined sets of incomplete random variables. Here, random variables are termed incomplete since they admit missing data. Two different notions of implication arise. In the classic notion of V -implication, a statement is implied jointly by a set of statements and a fixed set V of random variables. In the alternative notion of pure implication, a statement is implied by a given set of statements alone, leaving the set of random variables undetermined. A first axiomatization for V -implication is established that distinguishes purely implied from V -implied statements. Axiomatic, algorithmic and logical characterizations of pure implication are established. Pure implication appeals to applications in which the existence of random variables is uncertain, for example, when independence statements are integrated from different sources, when random variables are unknown or shall remain hidden.

1 INTRODUCTION

The concept of conditional independence (CI) is important for capturing structural aspects of probability distributions, for dealing with knowledge and uncertainty in artificial intelligence, and for learning and reasoning in intelligent systems [Darwiche (2009); Dawid (1979); Pearl (1988)]. Application areas include natural language processing, speech processing, computer vision, robotics, computational biology, and error-control coding [Darwiche (2009); Halpern (2005); Niepert et al. (2013)]. Central to these applications is the implication problem, which is to decide for an arbitrary set V of random variables, and an arbitrary set $\Sigma \cup \{\varphi\}$ of CI statements over V , whether every probability model that satisfies every element in Σ also sat-

isfies φ . Indeed, non-implied CI statements represent new opportunities to construct complex probability models with polynomially many parameters and to efficiently organize distributed probability computations [Geiger and Pearl (1993)]. An algorithm for deciding the implication problem can also test the consistency of independence and dependence statements collected from different sources; which is particularly important as these statements often introduce non-linear constraints resulting in unfeasible CSP instances [Geiger and Pearl (1993); Niepert et al. (2013)]. While the decidability of the implication problem for CI statements relative to discrete probability measures remains open, it is not axiomatizable by a finite set of Horn rules [Studený (1992)] and already coNP-complete for stable CI statements [Niepert, Van Gucht, and Gyssens (2010)]. An important subclass are therefore saturated CI (SCI) statements, in which all given random variables occur. In fact, graph separation and SCI statements enjoy the same axioms [Geiger and Pearl (1993)], and the implication problem of SCI statements is decidable in almost linear time [Galil (1982)]. These results contribute to the success story of Bayesian networks in AI and machine learning [Darwiche (2009); Geiger and Pearl (1993)], and have recently been carried over to the presence of missing data [Link (2013a)]. Here, independence is not judged on conditions that carry missing data. The findings complement a long line of AI research on the recognized need to reveal missing data and to explain where they come from, e.g. [Chickering and Heckerman (1997); Dempster, Laird, and Rubin (1977); Friedman (1997); Lauritzen (1995); Marlin et al. (2011); Saar-Tsechansky and Provost (2007); Singh (1997); Zhu et al. (2007)]. It is important to realize that implication problems of SCI statements in the presence of missing data differ from implication problems in the absence of data. For an illustration, consider a simplified burglary example. A $r(obbery)$ sets off an $a(larm)$ causing $s(heldon)$ or $b(atman)$ to call security. The independence between sb and r , given a , can be stated as the SCI statement $I(sb, r|a)$ over $V = \{b, a, r, s\}$. In the absence of missing data, $I(s, b|ar)$ and $I(sb, r|a)$ together do V -imply $I(s, br|a)$. With missing data present, however, $I(s, b|ar)$

and $I(sb, r|a)$ together do not V -imply $I(s, br|a)$:

r	a	b	s	P
—	true	true	true	0.5
—	true	false	false	0.5

Here, $I(s, b|ar)$ is satisfied as the assignments on the condition ar involve missing data, represented by —.

Most of the literature on the implication problem for SCI statements have focused on the notion of implication in which the underlying set V of random variables is assumed to be fixed. However, the assumption that V is fixed may not be practical: for example, the fact that not all random variables are known yet should not prevent us from declaring some independence statements; or even if we know all random variables, we may not want to disclose all of them; or when independence statements are integrated from different sources. Instead, we may want to state that given a , sb is independent from the set of remaining random variables, no matter what they are. This statement could be written as $I(sb|a)$. The intriguing point here is the difference between declaring $I(sb|a)$ and declaring $I(r|a)$ when V is left undetermined. In fact, the probability model

r	a	b	s	e	P
true	true	—	—	true	0.5
false	true	—	—	false	0.5

satisfies $I(sb|a)$, but does not satisfy $I(r|a)$. We conclude that $I(sb|a)$ implies $I(r|a)$ for the fixed set V , but $I(sb|a)$ does not imply $I(r|a)$ when the set of random variables is left undetermined.

The example illustrates the need to distinguish between different notions of semantic implication. The first notion is that of V -implication. For example, Link (2013a) established an axiomatization \mathfrak{U}_V for the V -implication problem of SCI statements in the presence of missing data. The alternative, stronger notion of pure implication leaves the set of random variables undetermined: the pure implication problem is to decide for every given set $\Sigma \cup \{\varphi\}$ of SCI statements, whether for every probability model π that involves at least all the random variables in $\Sigma \cup \{\varphi\}$ and that satisfies Σ , π also satisfies φ . Pure implication allows us to use independence statements without knowing all the random variables. This lowers barriers for their use and makes them applicable in demanding frameworks where some variables shall remain unknown for some users and where we still want to know how complex probability distributions can be organized efficiently. That is, pure implication enables us to reason under uncertainty about the random variables, while V -implication does not. For illustration, suppose we want to keep the random variable r hidden. Then it is impossible to reason about SCI statements under the notion of V -implication. With pure implication we can still state $I(sb|a)$ and $I(b|a)$, and our results show that we can even conclude $I(s|a)$ from that.

Contribution. In Section 2 we show that the only existing finite axiomatization \mathfrak{U}_V for the V -implication of SCI statements cannot distinguish between purely implied and V -implied SCI statements. That is, there are purely implied SCI statements for which every inference by \mathfrak{U}_V applies the V -symmetry rule; giving incorrectly the impression that the pure implication of an SCI statement depends on V . In Section 3 we establish a finite axiomatization \mathfrak{C}_V such that every purely implied SCI statement can be inferred without any application of the V -symmetry rule; every V -implied SCI statement can be inferred with only a single application of the V -symmetry rule, and this application is done in the last step of the inference. In Section 4 we establish a finite axiomatization \mathfrak{C} for the pure implication of SCI statements. As \mathfrak{C} results from \mathfrak{C}_V by removal of the symmetry rule, the results show that the symmetry rule is only necessary to infer those SCI statements that are V -implied but not implied. In Section 5, pure implication is characterized by V -implication where V involves random variables that do not occur in any of the given SCI statements. In Sections 6, 7 and 8 this result is exploited to characterize the pure implication problem i) logically by a propositional fragment under interpretations by Levesque’s situations, ii) by multivalued database dependencies involving missing data, and iii) by an algorithm that decides pure implication in almost linear time. Related work is discussed in Section 9. We conclude in Section 10.

2 IMPLICATION UNDER FIXED SETS OF RANDOM VARIABLES

We summarize the semantics of CI statements in the presence of missing data from Link (2013a). A definition is given that embodies the ability of an axiomatization to separate V -implied from purely implied SCI statements. It is shown that the existing axiomatization \mathfrak{U}_V for V -implication from Link (2013a) does not have this ability.

We denote by \mathfrak{V} a countably infinite set of distinct symbols $\{v_1, v_2, \dots\}$ of *random variables*. A *domain mapping* is a mapping that associates a set, $dom(v_i)$, with each random variable v_i of a finite set $V \subseteq \mathfrak{V}$. This set is called the *domain* of v_i and each of its elements is a *data value* of v_i . We assume that each domain $dom(v_i)$ contains the element —, which we call the *marker*. Although we use the element — like any other data value, we prefer to think of — as a marker, denoting that no information is currently available about the data value of v_i . The interpretation of this marker as no information means that a data value does either not exist (known as a structural zero in statistics, and the null marker inapplicable in databases), or a data value exists but is currently unknown (known as a sampling zero in statistics, and the null marker applicable in databases). The disadvantage of using this interpretation is a loss in knowledge when representing data values known to not

exist, or known to exist but currently unknown. This interpretation overcomes the computational difficulties when more expressive interpretations of missing data are used. As another key advantage one can represent missing data values, even if it is unknown whether they do not exist, or exist but are currently unknown. Strictly speaking, we shall call such random variables *incomplete* as their data values may be missing. For simplicity, we continue to speak off random variables for the remainder of this paper, although we really do mean incomplete random variables. For $X = \{v_1, \dots, v_k\} \subseteq V$ we say that \mathbf{a} is an assignment of X , if $\mathbf{a} \in \text{dom}(v_1) \times \dots \times \text{dom}(v_k)$. For an assignment \mathbf{a} of X we write $\mathbf{a}(y)$ for the projection of \mathbf{a} onto $Y \subseteq X$. We say that $\mathbf{a} = (\mathbf{a}_1, \dots, \mathbf{a}_k)$ is *X-complete*, if $\mathbf{a}_i \neq -$ for all $i = 1, \dots, k$.

A *probability model* over a finite set $V = \{v_1, \dots, v_n\}$ of random variables is a pair (dom, P) where dom is a domain mapping that maps each v_i to a finite domain $\text{dom}(v_i)$, and $P : \text{dom}(v_1) \times \dots \times \text{dom}(v_n) \rightarrow [0, 1]$ is a probability distribution having the Cartesian product of these domains as its sample space.

The expression $I(Y, Z|X)$ where X, Y and Z are disjoint subsets of V is called a *conditional independence (CI) statement* over V . The set X is called the *condition* of $I(Y, Z|X)$. If $XYZ = V$, we call $I(Y, Z|X)$ a *saturated CI (SCI) statement*. Let (dom, P) be a probability model over V . Following Link (2013a), a CI statement $I(Y, Z|X)$ is said to *hold for* (dom, P) if for every complete assignment \mathbf{x} of X , and for every assignment \mathbf{y}, \mathbf{z} of Y and Z , respectively,

$$P(\mathbf{x}, \mathbf{y}, \mathbf{z}) \cdot P(\mathbf{x}) = P(\mathbf{x}, \mathbf{y}) \cdot P(\mathbf{x}, \mathbf{z}). \quad (1)$$

Equivalently, (dom, P) is said to *satisfy* $I(Y, Z|X)$.

The satisfaction of $I(Y, Z|X)$ requires Equation 1 to hold for *complete* assignments \mathbf{x} of X only. The reason is that the independence between an assignment \mathbf{y} and an assignment \mathbf{z} is conditional on the assignment \mathbf{x} . Indeed, in case there is *no information* about the assignment \mathbf{x} , then there should not be any requirement on the independence between \mathbf{y} and \mathbf{z} .

SCI statements interact with one another, and these interactions have been formalized by the following notion of semantic implication. Let $\Sigma \cup \{\varphi\}$ be a set of SCI statements over V . We say that Σ *V-implies* φ , denoted by $\Sigma \models_V \varphi$, if every probability model over V that satisfies every SCI statement $\sigma \in \Sigma$ also satisfies φ . The *V-implication problem* is the following problem.

PROBLEM: V-implication problem	
INPUT:	Set V of random variables Set $\Sigma \cup \{\varphi\}$ of SCI statements over V
OUTPUT:	Yes, if $\Sigma \models_V \varphi$; No, otherwise

For Σ we let $\Sigma_V^* = \{\varphi \mid \Sigma \models_V \varphi\}$ be the *semantic closure*

Table 1: Axiomatization \mathcal{U} under Incomplete RVs

$\frac{}{I(V - X, \emptyset X)}$ (triviality, \mathcal{T}')	$\frac{I(Y, Z X)}{I(Z, Y X)}$ (symmetry, \mathcal{S})
$\frac{I(YZ, UW X) \ I(YU, ZW X)}{I(YZU, W X)}$ (algebra, \mathcal{A}')	$\frac{I(Y, ZW X)}{I(Y, Z XW)}$ (weak union, \mathcal{W}')

of Σ , i.e., the set of all SCI statements V -implied by Σ . In order to determine the V -implied SCI statements we use a syntactic approach by applying inference rules. These inference rules have the form

$$\frac{\text{premises}}{\text{conclusion}}$$

and inference rules without any premises are called axioms. An inference rule is called *V-sound*, if the premises of the rule V -imply the conclusion of the rule. We let $\Sigma \vdash_{\mathfrak{R}} \varphi$ denote the *inference* of φ from Σ by the set \mathfrak{R} of inference rules. That is, there is some sequence $\gamma = [\sigma_1, \dots, \sigma_n]$ of SCI statements such that $\sigma_n = \varphi$ and every σ_i is an element of Σ or results from an application of an inference rule in \mathfrak{R} to some elements in $\{\sigma_1, \dots, \sigma_{i-1}\}$. For Σ , let $\Sigma_{\mathfrak{R}}^+ = \{\varphi \mid \Sigma \vdash_{\mathfrak{R}} \varphi\}$ be its *syntactic closure* under inferences by \mathfrak{R} . A set \mathfrak{R} of inference rules is said to be *V-sound* (*V-complete*) for the V -implication of SCI statements, if for every V and for every set Σ of SCI statements over V , we have $\Sigma_{\mathfrak{R}}^+ \subseteq \Sigma_V^*$ ($\Sigma_V^* \subseteq \Sigma_{\mathfrak{R}}^+$). The (finite) set \mathfrak{R} is said to be a (finite) *axiomatization* for the V -implication of SCI statements if \mathfrak{R} is both V -sound and V -complete.

Table 1 contains the set $\mathcal{U} = \{\mathcal{T}', \mathcal{S}, \mathcal{A}', \mathcal{W}'\}$ of inference rules that form a finite axiomatization for the V -implication of SCI statements under incomplete random variables, as established in Link (2013a).

Motivated by the introductory remarks we now write $I(Y|X)$ instead of writing $I(V - XY, Y|X)$ for an SCI statement over V . It is first shown that the system $\mathcal{U}_V = \{\mathcal{T}, \mathcal{S}, \mathcal{A}, \mathcal{W}\}$ from Table 2 forms a finite axiomatization for the V -implication of such SCI statements under incomplete random variables.

Proposition 1 \mathcal{U}_V is a finite axiomatization for the V -implication of SCI statements under incomplete random variables.

Proof Let $V \subseteq \mathfrak{V}$ be a finite set of random variables. Let $\Sigma = \{I(Y_1|X_1), \dots, I(Y_n|X_n)\}$ and $\varphi = I(Y|X)$ be a (set of) SCI statement(s) over V . We can show by an induction over the inference length that $\Sigma \vdash_{\mathcal{U}_V} \varphi$ if

Table 2: Axiomatization \mathfrak{U}_V under Incomplete RVs

$\frac{}{I(\emptyset X)}$ (triviality, \mathcal{T})	$\frac{I(Y X)}{I(V - XY X)}$ (V-symmetry, \mathcal{S}_V)
$\frac{I(Y X) \quad I(Z X)}{I(YZ X)}$ (union, \mathcal{U})	$\frac{I(Y X)}{I(Y - Z XZ)}$ (weak union, \mathcal{W})

and only if $\Sigma' = \{I(Y_1, V - X_1 Y_1 | X_1), \dots, I(Y_n, V - X_n Y_n | X_n)\} \vdash_{\mathfrak{U}} I(V - XY, Y | X)$. Hence, the V -soundness (V -completeness) of \mathfrak{U}_V follows from the the V -soundness (V -completeness) of \mathfrak{U} . ■

Example 2 Consider $\Sigma = \{I(sb|a), I(b|a)\}$ and $\varphi = I(s|a)$ as a (set of) SCI statement(s) over $V = \{b, a, r, s\}$. Then $\Sigma \models_V \varphi$ as we can show, for example, by the following inference:

$$\frac{\frac{I(sb|a)}{\mathcal{S}_V : I(r|a)} \quad I(b|a)}{\mathcal{U} : I(rb|a)} \quad \mathcal{S}_V : I(s|a)$$

However, since the inference applies the V -symmetry rule it is not clear whether φ is implied by Σ alone, that is, whether it is true that for all V' that include at least a, s, b it holds that $\Sigma \models_{V'} \varphi$. In fact, if we were to find an inference of φ from Σ by \mathfrak{U}_V that never applies the V -symmetry rule \mathcal{S}_V , then we would know that φ is not only V -implied by Σ but even implied by Σ alone.

The last example motivates the following definition. It addresses the property of an inference system to first infer all those SCI statements implied by a set of SCI statements alone, without any application of the symmetry rule, and, subsequently, apply the V -symmetry rule once to some of these SCI statements to infer all V -implied SCI statements that do depend on the underlying set V of random variables.

Definition 3 Let \mathfrak{S}_V denote a set of inference rules that is V -sound for the V -implication of SCI statements, and in which the V -symmetry rule \mathcal{S}_V is the only inference rule that is dependent on V . We say that \mathfrak{S}_V is conscious of pure implication, if for every V , and every set $\Sigma \cup \{\varphi\}$ of SCI statements over V such that φ is V -implied by Σ there is some inference of φ from Σ by \mathfrak{S}_V such that the V -symmetry rule \mathcal{S}_V is applied at most once, and, if it is applied, then it is applied in the last step of the inference only.

Example 2 and Definition 3 motivate the question if \mathfrak{U}_V is conscious of pure implication.

Theorem 4 \mathfrak{U}_V is not conscious of pure implication.

Proof Let $V = \{b, a, r, s\}$ and $\Sigma = \{I(b|a), I(bs|a)\}$. One can show that $I(s|a) \notin \Sigma_{\{\mathcal{T}, \mathcal{W}, \mathcal{U}\}}^+$. Moreover, for all Y such that $r \in Y$, $I(Y|a) \notin \Sigma_{\{\mathcal{T}, \mathcal{W}, \mathcal{U}\}}^+$, see Lemma 10 from Section 4. However, $I(s|a) \in \Sigma_{\mathfrak{U}_V}^+$ as shown in Example 2. Consequently, in any inference of $I(s|a)$ from Σ by \mathfrak{U}_V the V -symmetry rule \mathcal{S}_V must be applied at least once, but is not just applied in the last step as $r \in V - \{b, a, s\}$. ■

In view of Theorem 4 it is natural to ask whether there is any axiomatization that is conscious of pure implication.

3 GAINING CONSCIOUSNESS

Theorem 4 has shown that axiomatizations are, in general, not conscious of pure implication. We will now establish a finite conscious axiomatization for the V -implication of SCI statements under incomplete random variables. For this purpose, we consider the difference rule \mathcal{D} as a new V -sound inference rule:

$$\frac{I(Y|X) \quad I(Z|X)}{I(Y - Z|X)}$$

The V -soundness of the difference rule \mathcal{D} follows easily from the algebra rule \mathcal{A}' .

Theorem 5 Let Σ be a set of SCI statements over V . For every inference γ from Σ by the system $\mathfrak{U}_V = \{\mathcal{T}, \mathcal{S}_V, \mathcal{U}, \mathcal{W}\}$ there is an inference ξ from Σ by the system $\mathfrak{C}_V = \{\mathcal{T}, \mathcal{S}_V, \mathcal{U}, \mathcal{W}, \mathcal{D}\}$ such that

1. γ and ξ infer the same SCI statement,
2. \mathcal{S}_V is applied at most once in ξ ,
3. if \mathcal{S}_V is applied in ξ , then as the last rule.

Proof The proof is done by induction on the length l of γ . For $l = 1$, the statement $\xi := \gamma$ has the desired properties. Suppose for the remainder of the proof that $l > 1$, and let $\gamma = [\sigma_1, \dots, \sigma_l]$ be an inference of σ_l from Σ by \mathfrak{U}_V . We distinguish between four different cases according to how σ_l is obtained from $[\sigma_1, \dots, \sigma_{l-1}]$.

Case 1. σ_1 is obtained from the triviality axiom \mathcal{T} , or is an element of Σ . In this case, $\xi := [\sigma_l]$ has the desired properties.

Case 2. We obtain σ_l by an application of the weak union rule \mathcal{W} to a premise σ_i with $i < l$. Let ξ_i be obtained by applying the induction hypothesis to $\gamma_i = [\sigma_1, \dots, \sigma_i]$. Consider the inference $\xi := [\xi_i, \sigma_l]$. If in ξ_i the V -symmetry rule \mathcal{S}_V is not applied, then ξ has the desired properties. If in ξ_i the \mathcal{S}_V is applied as the last rule, then the last two steps in ξ are of the following form:

$$\frac{\frac{I(Y|X)}{\mathcal{S}_V : I(V - XY|X)}}{\mathcal{W} : I(V - XYZ|XZ)}$$

However, these steps can be replaced as follows:

$$\frac{\frac{I(Y|X)}{W : I(Y - Z|XZ)}}{S_V : I(V - XYZ|XZ)} .$$

The resulting inference has the desired properties.

Case 3. We obtain σ_l by an application of the union rule \mathcal{U} to premises σ_i and σ_j with $i, j < l$. Let ξ_i and ξ_j be obtained by applying the induction hypothesis to $\gamma_i = [\sigma_1, \dots, \sigma_i]$ and $\gamma_j = [\sigma_1, \dots, \sigma_j]$, respectively. Consider the inference $\xi := [\xi_i, \xi_j, \sigma_l]$. We distinguish between four cases according to the occurrence of the V -symmetry rule S_V in ξ_i and ξ_j .

Case 3.1. If S_V does not occur in ξ_i nor in ξ_j , then ξ has the desired properties.

Case 3.2. If S_V occurs in ξ_i as the last rule but does not occur in ξ_j , then the last step of ξ_i and the last step of ξ are of the following form:

$$\frac{\frac{I(Y|X)}{S_V : I(V - XY|X)}}{U : \frac{I(Z|X)}{I((V - XY)Z|X)}} .$$

However, these steps can be replaced as follows:

$$\frac{\frac{I(Y|X)}{D : I(Y - Z|X)}}{S_V : I(V - ((Y - Z)X)|X)} .$$

$= (V - XY)Z$

The resulting inference has the desired properties.

Case 3.3. If S_V occurs in ξ_j as the last rule but does not occur in ξ_i , then the last step of ξ_j and the last step of ξ are of the following form:

$$\frac{\frac{I(Y|X)}{U : I((V - XZ)Y|X)}}{S_V : I(V - XZ|X)} .$$

However, these steps can be replaced as follows:

$$\frac{\frac{I(Z|X)}{D : I(Z - Y|X)}}{S_V : I(V - ((Z - Y)X)|X)} .$$

$= (V - XZ)Y$

The resulting inference has the desired properties.

Case 3.4. If S_V occurs in ξ_i as the last rule and occurs in ξ_j as the last rule, then the last steps of ξ_i and ξ_j and the last step of ξ are of the following form:

$$\frac{\frac{I(Y|X)}{S_V : I(V - XY|X)}}{U : \frac{I(Z|X)}{I((V - XY)(V - XZ)|X)}} .$$

However, these steps can be replaced as follows:

$$\frac{\frac{I(Y|X)}{D : \frac{I(Y - Z|X)}{I(Y - (Y - Z)|X)}}}{S_V : \frac{I(V - ((Y \cap Z)X)|X)}{= (V - XY)(V - XZ)}} .$$

The resulting inference has the desired properties.

Case 4. We obtain σ_l by an application of the V -symmetry rule S_V to a premise σ_i with $i < l$. Let ξ_i be obtained by applying the induction hypothesis to $\gamma_i = [\sigma_1, \dots, \sigma_i]$. Consider the inference $\xi := [\xi_i, \sigma_l]$. If in ξ_i the V -symmetry rule S_V is not applied, then ξ has the desired properties. If in ξ_i the V -symmetry rule S_V is applied as the last rule, then the last two steps in ξ are of the following form.

$$\frac{\frac{I(Y|X)}{S_V : I(V - XY|X)}}{S_V : I(V - (V - XY)X|X)} .$$

$= Y$

The inference obtained from deleting these steps has the desired properties. ■

Example 6 Recall Example 2 where $V = \{b, a, r, s\}$, $\Sigma = \{I(sb|a), I(b|a)\}$ and $\varphi = I(s|a)$. While the inference of φ from Σ using \mathcal{U}_V in Example 2 showed that $\Sigma \models_V \varphi$ holds, it did leave open the question whether Σ purely implies φ . Indeed, no inference of φ from Σ by \mathcal{U}_V can provide this insight by Theorem 4. However, using \mathcal{C}_V we can obtain the following inference of φ from Σ :

$$\frac{I(sb|a)}{D : \frac{I(b|a)}{I(s|a)}} .$$

Indeed, the V -symmetry rule S_V is unnecessary to infer φ from Σ .

Examples 2 and 6 indicate that the implication of $I(s|a)$ by $\Sigma = \{I(sb|a), I(b|a)\}$ does not depend on the fixed set V of random variables. In what follows we will formalize the stronger notion of pure implication as motivated in the introduction. Theorem 5 shows that the set $\mathcal{C} := \mathcal{C}_V - \{S_V\}$ of inference rules is nearly V -complete for the V -implication of SCI statements under incomplete random variables.

Theorem 7 Let $\Sigma \cup \{I(Y|X)\}$ be a set of SCI statements over the set V of incomplete random variables. Then $I(Y|X) \in \Sigma_{\mathcal{C}_V}^+$ if and only if $I(Y|X) \in \Sigma_{\mathcal{C}}^+$ or $I(V - XY|X) \in \Sigma_{\mathcal{C}}^+$. ■

Theorem 7 indicates that \mathcal{C} can infer every implied SCI statement that is independent from the set V of incomplete

random variables. Another interpretation of Theorem 7 is the following. In using \mathfrak{C} to infer V -implied statements, the fixation of V can be deferred until the last step of an inference.

4 PURE IMPLICATION

In this section we formalize the notion of pure implication as motivated in the introduction. It is shown that the set \mathfrak{C} of inference rules forms a finite axiomatization for pure implication. On the one hand, this allows us to distinguish between V -implied and purely implied statements. On the other hand, the notion of pure implication can be applied whenever this notion of implication is more convenient to use, for examples, when there is uncertainty about additional random variables that may be required in the future, when some variables are unknown, or when some variables are meant to remain hidden.

A *probability model* is a triple (V, dom, P) where $V = \{v_1, \dots, v_n\} \subseteq \mathfrak{V}$ is a finite set of incomplete random variables, dom is a domain mapping that maps each v_i to a finite domain $\text{dom}(v_i)$, and $P : \text{dom}(v_1) \times \dots \times \text{dom}(v_n) \rightarrow [0, 1]$ is a probability distribution having the Cartesian product of these domains as its sample space. The expression $I(Y|X)$ where X and Y are finite, disjoint subsets of \mathfrak{V} is called a *saturated conditional independence (SCI) statement*. We say that the SCI statement $I(Y|X)$ *holds for* (V, dom, P) if $XY \subseteq V$ and for every complete assignment \mathbf{x} of X , every assignment \mathbf{y} of Y , and every assignment \mathbf{z} of $V - XY$, respectively,

$$P(\mathbf{x}, \mathbf{y}, \mathbf{z}) \cdot P(\mathbf{x}) = P(\mathbf{x}, \mathbf{y}) \cdot P(\mathbf{x}, \mathbf{z}).$$

Equivalently, (V, dom, P) is said to *satisfy* $I(Y|X)$. For an SCI statement $\sigma = I(Y|X)$ let $V_\sigma := XY$, and for a finite set Σ of SCI statements let $V_\Sigma := \bigcup_{\sigma \in \Sigma} V_\sigma$ denote the random variables that occur in it.

Definition 8 Let $\Sigma \cup \{\varphi\}$ be a finite set of SCI statements. We say that Σ *purely implies* φ , denoted by $\Sigma \models \varphi$, if and only if every probability model (V, dom, P) with $V_{\Sigma \cup \{\varphi\}} \subseteq V$ that satisfies every SCI statement $\sigma \in \Sigma$ also satisfies φ .

In the definition of pure implication the set of incomplete random variables is left undetermined. The only requirement is that the SCI statements must apply to the probability model. The pure implication problem for SCI statements can be stated as follows.

PROBLEM:	Pure Implication Problem
INPUT:	Set $\Sigma \cup \{\varphi\}$ of SCI statements
OUTPUT:	Yes, if $\Sigma \models \varphi$; No, otherwise

Pure implication is stronger than V -implication.

Table 3: Axiomatization \mathfrak{C} for Pure Implication

$\overline{I(\emptyset X)}$ (triviality, \mathcal{T})	$\overline{I(Y X)}$ $\overline{I(Y - Z XZ)}$ (weak union, \mathcal{W})
$\frac{I(Y X) \quad I(Z X)}{I(YZ X)}$ (union, \mathcal{U})	$\frac{I(Y X) \quad I(Z X)}{I(Y - Z X)}$ (difference, \mathcal{D})

Proposition 9 Let $\Sigma \cup \{\varphi\}$ be a finite set of SCI statements, such that $V_{\Sigma \cup \{\varphi\}} \subseteq V$. If $\Sigma \models \varphi$, then $\Sigma \models_V \varphi$, but the other direction may fail.

Proof The first statement follows directly from the definitions of pure and V -implication. For the other direction, let $V = \{b, a, r, s\}$, $\Sigma = \{I(r|a)\}$ and let φ be $I(sb|a)$. Clearly, Σ V -implies φ . However, Σ does not purely imply φ as the example from the introduction shows. ■

Soundness and completeness for pure implication are defined as their corresponding notions in the context of some fixed set V by dropping the reference to V . While triviality axiom \mathcal{T} , weak union rule \mathcal{W} , and union rule \mathcal{U} are all sound, the V -symmetry rule \mathcal{S}_V is V -sound but not sound.

We shall now prove that \mathfrak{C} forms a finite axiomatization for the pure implication of SCI statements. For this purpose, we prove two lemmata in preparation. The correctness of the first lemma can easily be observed by inspecting the inference rules in \mathfrak{C} . For each of the rules, every random variable that occurs on the left-hand side of the bar in the conclusion of the rule, already appears on the left-hand side of the bar in at least one premise of the rule.

Lemma 10 Let $\Sigma = \{I(Y_1|X_1), \dots, I(Y_n|X_n)\}$ be a finite set of SCI statements. If $I(Y|X) \in \Sigma_{\mathfrak{C}}^+$, then $Y \subseteq Y_1 \cup \dots \cup Y_n$. ■

For the next lemma one may notice that the random variables that do not occur in V_Σ can always be introduced in the last step of an inference, by applying the weak union rule \mathcal{W} .

Lemma 11 Let Σ be a finite set of SCI statements. If $I(Y|X) \in \Sigma_{\mathfrak{C}}^+$, then there is an inference $\gamma = [\sigma_1, \dots, \sigma_l]$ of $I(Y|X)$ from Σ by \mathfrak{C} such that every attribute occurring in $\sigma_1, \dots, \sigma_{l-1}$ is an element of V_Σ .

Proof Define $W := V_\Sigma$ and let $\bar{\xi} := [I(V_1|U_1), \dots, I(V_{l-1}|U_{l-1})]$ be an inference of $I(Y|X)$ from Σ by \mathfrak{C} . Consider the sequence

$$\xi := [I(V_1 \cap W|U_1 \cap W), \dots, I(V_{l-1} \cap W|U_{l-1} \cap W)].$$

We claim that ξ is an inference of $I(Y \cap W | X \cap W)$ from Σ by \mathcal{C} . For if $I(V_i | U_i)$ is an element of Σ or was obtained by an application of the triviality axiom \mathcal{T} , then $I(Y \cap W | X \cap W) = I(Y | X)$. One can verify that if $I(V_i | U_i)$ is the result of applying one of the rules $\mathcal{U}, \mathcal{W}, \mathcal{D}$, then $I(V_i \cap W | U_i \cap W)$ is the result of the same rule applied to the corresponding premises in ξ .

Now by Lemma 10 we know that $Y \subseteq W$, hence $Y \cap W = Y$. However, this means that we can infer $I(Y | X)$ from $I(Y \cap W | X \cap W)$ by a single application of the weak union rule \mathcal{W} :

$$\frac{I(Y \cap W | X \cap W)}{I(\underbrace{(Y \cap W) - X}_{=Y} | \underbrace{(X \cap W) \cup X}_{=X})}.$$

Hence, the inference $[\xi, I(Y | X)]$ has the desired properties. ■

We are now prepared to prove the following result.

Theorem 12 *The set $\mathcal{C} = \{\mathcal{T}, \mathcal{W}, \mathcal{U}, \mathcal{D}\}$ forms a finite axiomatization for the pure implication of SCI statements under incomplete random variables.*

Proof Let $\Sigma = \{I(Y_1 | X_1), \dots, I(Y_n | X_n)\}$ be a finite set of SCI statements and $I(Y | X)$ an SCI statement. We have to show that

$$I(Y | X) \in \Sigma^* \quad \text{if and only if} \quad I(Y | X) \in \Sigma_{\mathcal{C}}^+.$$

Let $T := X \cup Y \cup V_{\Sigma}$. In order to prove the soundness of \mathcal{C} we assume that $I(Y | X) \in \Sigma_{\mathcal{C}}^+$ holds. Let (V, dom, P) be a probability model that satisfies every element of Σ , and where $T \subseteq V$ holds. We must show that (V, dom, P) also satisfies $I(Y | X)$. According to Lemma 11 there is an inference γ of $I(Y | X)$ from Σ by \mathcal{C} such that $U \cup W \subseteq T \subseteq V$ holds for each SCI statement $I(W | U)$ that occurs in γ . Since each rule in \mathcal{C} is sound we can conclude (by induction) that each SCI statement occurring in γ is satisfied by (V, dom, P) . In particular, (V, dom, P) satisfies $I(Y | X)$.

In order to prove the completeness of \mathcal{C} we assume that $I(Y | X) \notin \Sigma_{\mathcal{C}}^+$. Let $V \subseteq \mathfrak{V}$ be a finite set of random variables such that T is a proper subset of V , i.e., $T \subset V$. Consequently, $V - XY$ is not a subset of T . Hence, by Lemma 10, $I(V - XY | X) \notin \Sigma_{\mathcal{C}}^+$. Now from $I(Y | X) \notin \Sigma_{\mathcal{C}}^+$ and from $I(V - XY | X) \notin \Sigma_{\mathcal{C}}^+$ we conclude that $I(Y | X) \notin \Sigma_{\mathcal{C}_V}^+$ by Theorem 7. Since \mathcal{C}_V is V -complete for the V -implication of SCI statements it follows that Σ does not V -imply $I(Y | X)$. Hence, Σ does not purely imply $I(Y | X)$ by Proposition 9. ■

Example 13 *Recall Example 6 where $V = \{b, a, r, s\}$, and Σ consists of the two SCI statements $I(bs|a)$ and $I(b|a)$. The inference of $I(s|a)$ from Σ by \mathcal{C}_V in Example 6 is actually an inference by \mathcal{C} . Hence, $I(s|a)$ is purely implied by Σ , as one would expect intuitively.*

5 PURE AND V -IMPLICATION

Instances $\Sigma \models \varphi$ of the pure implication problem can be characterized by the instance $\Sigma \models_V \varphi$ of the V -implication problem for any set V of incomplete random variables that properly contains $V_{\Sigma \cup \{\varphi\}}$.

Theorem 14 *Let $\Sigma \cup \{\varphi\}$ be a set of SCI statements. Then the following are equivalent:*

1. $\Sigma \models \varphi$
2. for some V such that $V_{\Sigma \cup \{\varphi\}} \subset V$, $\Sigma \models_V \varphi$
3. for all V such that $V_{\Sigma \cup \{\varphi\}} \subset V$, $\Sigma \models_V \varphi$

Proof It is clear that 3. entails 2. Let $\varphi = I(Y | X)$, and let V be any finite set of random variables such that $V_{\Sigma \cup \{\varphi\}} \subset V$. If 2. holds, then Theorem 7 and Theorem 12 show that 1. holds or $\Sigma \vdash_{\mathcal{C}} I(V - XY | X)$ holds. However, Lemma 10 shows that the latter condition cannot hold as $V - XY$ contains some random variable that does not occur in V_{Σ} . Hence, 2. entails 1. If 1. holds, then Theorem 7 and Theorem 12 show that 3. holds as well. ■

Example 15 $\Sigma = \{I(bs|a), I(b|a)\}$ purely implies $I(s|a)$ as, for instance, $\Sigma \models_V I(s|a)$ for $V = \{b, a, r, s\}$. $\Sigma' = \{I(bs|a)\}$ does not purely imply $I(r|a)$ as for $V = \{b, e, a, r, s\}$, Σ' does not V -imply $I(r|a)$ as witnessed in the introduction.

In the following we apply Theorem 14 to establish characterizations of pure implication in terms of logical formulae under Levesque's situations, database dependencies, and algorithmic solutions. For a set $\Sigma \cup \{\varphi\}$ of SCI statements we write $V_c = V_{\Sigma \cup \{\varphi\}} \cup \{v_0\}$ for some $v_0 \notin V_{\Sigma \cup \{\varphi\}}$, $\sigma_c = I(V_c - XY, Y | X)$ for $\sigma = I(Y | X) \in \Sigma \cup \{\varphi\}$ and $\Sigma_c = \{\sigma_c \mid \sigma \in \Sigma\}$. In particular, $\Sigma \models \varphi$ if and only if $\Sigma_c \models_{V_c} \varphi_c$.

6 LEVESQUE'S SITUATIONS

We recall the framework for situations from Levesque (1989), and exploit them to establish a logical characterization of the pure implication problem.

For a finite set L of propositional variables, let L^* denote the *propositional language* over L , generated from the unary connective \neg (negation), and the binary connectives \wedge (conjunction) and \vee (disjunction). Elements of L^* are also called formulae of L , and usually denoted by φ', ψ' or their subscripted versions. Sets of formulae are denoted by Σ' . We omit parentheses if this does not cause ambiguity.

Let L^ℓ denote the set of all literals over L , i.e., $L^\ell = L \cup \{\neg v' \mid v' \in L\}$. A *situation* of L is a total function $\omega : L^\ell \rightarrow \{\mathbb{F}, \mathbb{T}\}$ that does not map both a propositional variable $v' \in L$ and its negation $\neg v'$ to \mathbb{F} . That is, we must not have $\omega(v') = \mathbb{F} = \omega(\neg v')$ for any $v' \in L$.

A situation $\omega : L^\ell \rightarrow \{\mathbb{F}, \mathbb{T}\}$ of L can be lifted to a total function $\Omega : L^* \rightarrow \{\mathbb{F}, \mathbb{T}\}$. Assuming φ' is in Negation Normal Form, this lifting is defined by:

- $\Omega(\varphi') = \omega(\varphi')$, if $\varphi' \in L^\ell$,
- $\Omega(\varphi' \vee \psi') = \mathbb{T}$ iff $\Omega(\varphi') = \mathbb{T}$ or $\Omega(\psi') = \mathbb{T}$,
- $\Omega(\varphi' \wedge \psi') = \mathbb{T}$ iff $\Omega(\varphi') = \mathbb{T}$ and $\Omega(\psi') = \mathbb{T}$.

A situation ω is a *model* of a set Σ' of L -formulae if and only if $\Omega(\sigma') = \mathbb{T}$ holds for every $\sigma' \in \Sigma'$. We say that Σ' *implies* an L -formula φ' , denoted by $\Sigma' \models_L \varphi'$, if and only if every situation that is a model of Σ' is also a model of φ' .

Equivalences. Let $\phi : V_c \rightarrow L_c$ denote a bijection between a set V_c of random variables and the set $L_c = \{v' \mid v \in V\}$ of propositional variables. We extend ϕ to a mapping Φ from the set of SCI statements over V_c to the set L_c^* . For an SCI statement $I(Y, Z \mid X)$ over V_c , let $\Phi(I(Y, Z \mid X))$ denote

$$\bigvee_{v \in X} \neg v' \vee \left(\bigwedge_{v \in Y} v' \right) \vee \left(\bigwedge_{v \in Z} v' \right).$$

Disjunctions over zero disjuncts are \mathbb{F} and conjunctions over zero conjuncts are \mathbb{T} . We will denote $\Phi(\varphi_c) = \varphi'_c$ and $\Phi(\Sigma_c) = \{\Phi(\sigma_c) \mid \sigma \in \Sigma_c\} = \Sigma'_c$.

In our example, for $\varphi_c = I(bse, r \mid a)$ we have $\varphi'_c = \neg a' \vee (b' \wedge s' \wedge e') \vee r'$, and for $\Sigma_c = \{I(re, bs \mid a)\}$ we have $\Sigma'_c = \{\neg a' \vee (b' \wedge s' \wedge e') \vee (r' \wedge e')\}$.

It was shown in Link (2013a) that for any set $\Sigma_c \cup \{\varphi_c\}$ of SCI statements over V_c there is a probability model $\pi = (dom, P)$ over V_c that satisfies Σ_c and violates φ_c if and only if there is a situation ω_π over L_c that is a model of Σ'_c but not a model of φ'_c . For arbitrary probability models π it is not obvious how to define the situation ω_π . However, if Σ_c does not V_c -imply φ_c , then there is a special probability model $\pi = (dom, \{\mathbf{a}_1, \mathbf{a}_2\})$ over V_c that i) has two assignments $\mathbf{a}_1, \mathbf{a}_2$ of probability one half each, ii) satisfies all SCI statements in Σ_c and iii) violates φ_c . Given such π , let ω_π denote the following special situation of L_c , taken from Link (2013a):

$$\begin{aligned} \omega_\pi(v') &= \begin{cases} \mathbb{T} & , \text{ if } \mathbf{a}_1(v) = \mathbf{a}_2(v) \\ \mathbb{F} & , \text{ otherwise} \end{cases} \quad , \text{ and} \\ \omega_\pi(\neg v') &= \begin{cases} \mathbb{T} & , \text{ if } \mathbf{a}_1(v) = \mu = \mathbf{a}_2(v) \text{ or} \\ & \mathbf{a}_1(v) \neq \mathbf{a}_2(v) \\ \mathbb{F} & , \text{ otherwise} \end{cases} . \end{aligned}$$

From the results in Link (2013a) and Theorem 14 we obtain the following logical characterization of pure implication.

Theorem 16 *Let $\Sigma \cup \{\varphi\}$ be a finite set of SCI statements and $L_c = \{v' \mid v \in V_{\Sigma \cup \{\varphi\}} \cup \{v_0\}\}$. Then $\Sigma \models \varphi$ if and only if $\Sigma'_c \models_{L_c} \varphi'_c$.*

Proof Theorem 14 shows that $\Sigma \models \varphi$ if and only if

$\Sigma_c \models_{V_c} \varphi_c$ for $V_c = V_{\Sigma \cup \{\varphi\}} \cup \{v_0\}$. By (Link, 2013a, Thm.6), $\Sigma_c \models_{V_c} \varphi_c$ if and only if $\Sigma'_c \models_{L_c} \varphi'_c$. ■

Recall that $\Sigma = \{I(sb \mid a)\}$ does not purely imply $\varphi = I(s, br \mid a)$ as the special probability model π defined by

r	a	b	s	e	P
true	true	—	—	true	0.5
false	true	—	—	false	0.5

satisfies Σ_c , but violates φ_c . Any special situation where $\omega_\pi(b') = \mathbb{T} = \omega_\pi(s')$, $\omega_\pi(\neg a') = \omega_\pi(r') = \omega_\pi(e') = \mathbb{F}$ is a model of $\Sigma'_c = \{\neg a' \vee (b' \wedge s') \vee (r' \wedge e')\}$, but not a model of $\varphi'_c = \neg a' \vee (b' \wedge s' \wedge e') \vee r'$.

7 DATABASE DEPENDENCIES

Database dependencies enforce the semantics of application domains in database systems [Link (2001)]. Let $\mathfrak{A} = \{\hat{v}_1, \hat{v}_2, \dots\}$ be an infinite set of distinct symbols, called attributes. A *relation schema* is a finite non-empty subset R of \mathfrak{A} . Each attribute $\hat{v} \in R$ has an infinite domain $dom(\hat{v})$. In order to encompass missing data values the domain of each attribute contains the null marker $-$. The intention of $-$ is to mean “no information” [Lien (1982)]. A *tuple* over R is a function $t : R \rightarrow \bigcup_{\hat{v} \in R} dom(\hat{v})$ with $t(\hat{v}) \in dom(\hat{v})$ for all $\hat{v} \in R$. For $X \subseteq R$ let $t(X)$ denote the restriction of t to X . A *relation* r over R is a finite set of tuples over R . For a tuple t over R and a set $X \subseteq R$, t is said to be *X-total*, if for all $\hat{v} \in X$, $t(\hat{v}) \neq -$. A relation over R is a *total relation*, if it is R -total. A *multivalued dependency* (MVD) over R is a statement $X \twoheadrightarrow Y$ where X and Y are disjoint subsets of R [Lien (1982)]. The MVD $X \twoheadrightarrow Y$ over R is satisfied by a relation r over R if and only if for all $t_1, t_2 \in r$ the following holds: if t_1 and t_2 are X -total and $t_1(X) = t_2(X)$, then there is some $t \in r$ such that $t(XY) = t_1(XY)$ and $t(X(R - XY)) = t_2(X(R - XY))$. Thus, the relation r satisfies $X \twoheadrightarrow Y$ when every X -total value determines the set of values on Y independently of the set of values on $R - Y$. For a set $\hat{\Sigma} \cup \{\hat{\varphi}\}$ of MVDs over R , $\hat{\Sigma}$ R -implies $\hat{\varphi}$, denoted by $\hat{\Sigma} \models_R \hat{\varphi}$, if and only if every relation over R that satisfies all elements in $\hat{\Sigma}$ also satisfies $\hat{\varphi}$.

For a set $\Sigma_c \cup \{\varphi_c\}$ of SCI statements over V_c one may associate the set $\hat{\Sigma}_c \cup \{\hat{\varphi}_c\}$ of MVDs over $R_c := \{\hat{v} \mid v \in V_c\}$, where $\hat{\sigma}_c = X \twoheadrightarrow Y$ for $\sigma_c = I(Y, Z \mid X)$ and $\hat{\Sigma}_c = \{\hat{\sigma}_c \mid \sigma \in \Sigma_c\}$.

Theorem 17 *Let $\Sigma \cup \{\varphi\}$ be a finite set of SCI statement. Then $\Sigma \models \varphi$ if and only if $\hat{\Sigma}_c \models_{R_c} \hat{\varphi}_c$.*

Proof Theorem 14 shows that $\Sigma \models \varphi$ if and only if $\Sigma_c \models_{V_c} \varphi_c$ for $V_c = V_{\Sigma \cup \{\varphi\}} \cup \{v_0\}$. By (Link, 2013a, Thm.8), $\Sigma_c \models_{V_c} \varphi_c$ if and only if $\hat{\Sigma}_c \models_{R_c} \hat{\varphi}_c$. ■

8 ALGORITHM & COMPLEXITY

(Link, 2013a, Thm. 7) shows that $\Sigma_c \models_{V_c} \varphi_c$ for $\varphi_c = I(Z, Y|X)$ holds if and only if $\Sigma_c[X] \models_{V_c} \varphi_c$ holds classically, that is, when no domain contains the marker. Here, $\Sigma_c[X] := \{I(V, W|U) \mid I(V, W|U) \in \Sigma_c \wedge U \subseteq X\}$. The *independence basis* $IDepB_{\Sigma_c[X]}(X)$ consists of the minimal $Y \subseteq V_c - X$ such that $\Sigma_c[X] \models_{V_c} I(Z, Y|X)$. By (Link, 2013a, Thm. 8), $\Sigma \models \varphi$ if and only if $\Sigma_c[X] \models_{R_c} \hat{\varphi}_c$, that is, every total relation over R_c that satisfies $\Sigma_c[X]$ also satisfies $\hat{\varphi}_c$. Galil (1982) gave an efficient algorithmic solution to the latter problem.

Theorem 18 *Using the algorithm in Galil (1982), the pure implication problem $\Sigma \models I(Y|X)$ can be decided in time $\mathcal{O}(|\Sigma_c| + \min\{k_{\Sigma_c[X]}, \log \bar{p}_{\Sigma_c[X]}\} \times |\Sigma_c[X]|)$. Herein, $|\Sigma_c|$ denotes the total number of random variables in Σ_c , $k_{\Sigma_c[X]}$ denotes the cardinality of $\Sigma_c[X]$, and $\bar{p}_{\Sigma_c[X]}$ denotes the number of sets in $IDepB_{\Sigma_c[X]}(X)$ that have non-empty intersection with Y .* ■

9 RELATED WORK

Dawid (1979) first investigated fundamental properties of conditional independence, leading to a claim that “rather than just being another useful tool in the statistician’s kitbag, conditional independence offers a new language for the expression of statistical concepts and a framework for their study”. Geiger and Pearl (1993) have systematically investigated the implication problem for fragments of CI statements over different probability models. In particular, they have established an axiomatization of SCI statements by a finite set of Horn rules. Studený (1992) showed that no axiomatization by a finite set of Horn rules exists for general CI statements. Niepert, Van Gucht, and Gyssens (2010) established an axiomatization for stable CI statements, which subsume SCI statements, and showed that their associated implication problem is coNP-complete. Independently, database theory has investigated the concept of embedded multivalued dependencies (MVDs) whose implication problem is undecidable [Herrmann (1995)] and not axiomatizable by a finite set of Horn rules [Stott Parker Jr. and Parsaye-Ghomi (1980)]. Studený (1992) also showed that the implication problem of embedded MVDs and that of CI statements do not coincide. In contrast, the implication problems of MVDs, SCI statements and some fragment of Boolean propositional logic all coincide [Geiger and Pearl (1993); Sagiv et al. (1981); Wong, Butz, and Wu (2000)]. These findings have been established for the notion of implication over fixed sets of variables and the idealized case where all data values are known. Biskup, Hartmann, and Link (2012) differentiated between V -implication and pure implication for SCI statements with complete random variables only, applying ideas from database theory in Biskup (1980) and

Link (2012). In the case of missing data, equivalences between implication problems for MVDs with null markers, SCI statements with incomplete random variables, and a fragment of propositional logic under Levesque’s situations were established recently in Link (2013a) and Hartmann and Link (2012). However, the notion of pure implication for conditional independence statements has not been studied yet in the context of missing data.

10 CONCLUSION

Recently, probabilistic conditional independence statements were studied in the presence of incomplete random variables, which admit missing data values. The associated implication problem for saturated CI statements was characterized axiomatically by a finite set \mathcal{U}_V of Horn rules, logically by a propositional fragment under interpretations by Levesque’s situations, and algorithmically by an equivalence to database dependencies. In this paper it was shown that there is a difference between SCI statements V -implied jointly by a given set of SCI statements and a fixed set V of incomplete random variables, and those purely implied by a given set of SCI statements alone. It was shown that \mathcal{U}_V cannot separate V -implied from purely implied SCI statements. An axiomatization \mathcal{C}_V was then established that can infer any purely implied SCI statement without applications of the V -symmetry rule \mathcal{S}_V , and infer any V -implied SCI statement with a single application of \mathcal{S}_V in the very last step of the inference only. The system \mathcal{C} that results from \mathcal{C}_V by removing \mathcal{S}_V was proven to form a finite axiomatization for the stronger notion of pure implication. The pure implication problem $\Sigma \models \varphi$ was characterized by the V -implication problem $\Sigma \models_V \varphi$ for sets V that properly contain the random variables that occur $\Sigma \cup \{\varphi\}$. This result enabled us to characterize pure implication logically and algorithmically as well. Our results clarify the role of the V -symmetry rule \mathcal{S}_V as a pure means to infer V -implied SCI statements. The notion of pure implication is appealing when the existence of random variables is uncertain, for example, when independence statements are integrated from different sources, when random variables are unknown or when they shall remain hidden. It is future work to extend the findings of this paper to the general case where an arbitrary finite set S of complete random variables can be specified, thereby, covering the current setting by the case where $S = \emptyset$ and the classical setting by the case where every random variable is complete. This would extend the work in Link (2013b) where the notion of pure implication was not considered.

Acknowledgements

This research is supported by the Marsden fund council from Government funding, administered by the Royal Society of New Zealand.

References

- Biskup, J.; Hartmann, S.; and Link, S. 2012. Probabilistic conditional independence under schema certainty and uncertainty. In *SUM*, 365–378.
- Biskup, J. 1980. Inferences of multivalued dependencies in fixed and undetermined universes. *Theor. Comput. Sci.* 10(1):93–106.
- Chickering, D. M., and Heckerman, D. 1997. Efficient approximations for the marginal likelihood of bayesian networks with hidden variables. *Machine Learning* 29(2-3):181–212.
- Darwiche, A. 2009. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press.
- Dawid, A. P. 1979. Conditional independence in statistical theory. *Journal of the Royal Statistical Society. Series B (Methodological)* 41(1):1–31.
- Dempster, A.; Laird, N. M.; and Rubin, D. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B* 39:139.
- Friedman, N. 1997. Learning belief networks in the presence of missing values and hidden variables. In *ICML*, 125–133.
- Galil, Z. 1982. An almost linear-time algorithm for computing a dependency basis in a relational database. *J. ACM* 29(1):96–102.
- Geiger, D., and Pearl, J. 1993. Logical and algorithmic properties of conditional independence and graphical models. *The Annals of Statistics* 21(4):2001–2021.
- Halpern, J. 2005. *Reasoning about uncertainty*. MIT Press.
- Hartmann, S., and Link, S. 2012. The implication problem of data dependencies over SQL table definitions: axiomatic, algorithmic and logical characterizations. *ACM Trans. Database Syst.* 37(2):Article 13.
- Herrmann, C. 1995. On the undecidability of implications between embedded multivalued database dependencies. *Inf. Comput.* 122(2):221–235.
- Lauritzen, S. 1995. The EM algorithm for graphical association models with missing data. *Computational Statistics and Data Analysis* 19:191–201.
- Levesque, H. 1989. A knowledge-level account of abduction. In *IJCAI*, 1061–1067.
- Lien, E. 1982. On the equivalence of database models. *J. ACM* 29(2):333–362.
- Link, S. 2001. Consistency enforcement in databases. In *Semantics in Databases*, 139–159.
- Link, S. 2012. Characterizations of multivalued dependency implication over undetermined universes. *J. Comput. Syst. Sci.* 78(4):1026–1044.
- Link, S. 2013a. Reasoning about saturated conditional independence under uncertainty. In *AAAI*.
- Link, S. 2013b. Sound approximate reasoning about saturated conditional probabilistic independence under controlled uncertainty. *J. Applied Logic* 11(3):309–327.
- Marlin, B. M.; Zemel, R. S.; Roweis, S. T.; and Slaney, M. 2011. Recommender systems, missing data and statistical model estimation. In *IJCAI*, 2686–2691.
- Niepert, M.; Gyssens, M.; Sayrafi, B.; and Gucht, D. V. 2013. On the conditional independence implication problem: A lattice-theoretic approach. *Artif. Intell.* 202:29–51.
- Niepert, M.; Van Gucht, D.; and Gyssens, M. 2010. Logical and algorithmic properties of stable conditional independence. *Int. J. Approx. Reasoning* 51(5):531–543.
- Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, U.S.A.: Morgan Kaufmann.
- Saar-Tsechansky, M., and Provost, F. J. 2007. Handling missing values when applying classification models. *Journal of Machine Learning Research* 8:1623–1657.
- Sagiv, Y.; Delobel, C.; Parker Jr., D. S.; and Fagin, R. 1981. An equivalence between relational database dependencies and a fragment of propositional logic. *J. ACM* 28(3):435–453.
- Singh, M. 1997. Learning bayesian networks from incomplete data. In *AAAI/IAAI*, 534–539.
- Stott Parker Jr., D., and Parsaye-Ghomi, K. 1980. Inferences involving embedded multivalued dependencies and transitive dependencies. In *SIGMOD*, 52–57.
- Studený, M. 1992. Conditional independence relations have no finite complete characterization. In *11th Prague Conference on Information Theory, Statistical Decision Functions and Random Processes*, 377–396.
- Wong, S.; Butz, C.; and Wu, D. 2000. On the implication problem for probabilistic conditional independency. *Trans. Systems, Man, and Cybernetics, Part A: Systems and Humans* 30(6):785–805.
- Zhu, X.; Zhang, S.; Zhang, J.; and Zhang, C. 2007. Cost-sensitive imputing missing values with ordering. In *AAAI*, 1922–1923.

Matroid Bandits: Fast Combinatorial Optimization with Learning

Branislav Kveton, Zheng Wen, Azin Ashkan, Hoda Eydgahi, and Brian Eriksson

Technicolor Labs

Los Altos, CA

{branislav.kveton,zheng.wen,azin.ashkan,hoda.eydgahi,brian.eriksson}@technicolor.com

Abstract

A matroid is a notion of independence in combinatorial optimization which is closely related to computational efficiency. In particular, it is well known that the maximum of a constrained modular function can be found greedily if and only if the constraints are associated with a matroid. In this paper, we bring together the ideas of bandits and matroids, and propose a new class of combinatorial bandits, *matroid bandits*. The objective in these problems is to learn how to maximize a modular function on a matroid. This function is stochastic and initially unknown. We propose a practical algorithm for solving our problem, *Optimistic Matroid Maximization* (OMM); and prove two upper bounds, gap-dependent and gap-free, on its regret. Both bounds are sublinear in time and at most linear in all other quantities of interest. The gap-dependent upper bound is tight and we prove a matching lower bound on a partition matroid bandit. Finally, we evaluate our method on three real-world problems and show that it is practical.

1 Introduction

Combinatorial optimization is a well-established field that has many practical applications, ranging from resource allocation [14] to designing network routing protocols [20]. Modern combinatorial optimization problems are often so massive that even low-order polynomial-time solutions are not practical. Fortunately, many important problems, such as finding a minimum spanning tree, can be solved greedily. Such problems can be often viewed as optimization on a *matroid* [25], a notion of independence in combinatorial optimization which is closely related to computational efficiency. In particular, it is well known that the maximum of a constrained modular function can be found greedily if and only if all feasible solutions to the problem are the in-

dependent sets of a matroid [8]. Matroids are common in practice because they generalize many notions of independence, such as linear independence and forests in graphs.

In this paper, we propose an algorithm for learning how to maximize a stochastic modular function on a matroid. The modular function is represented as the sum of the weights of up to K items, which are chosen from the ground set E of a matroid, which has L items. The weights of the items are stochastic and represented as a vector $\mathbf{w} \in [0, 1]^L$. The vector \mathbf{w} is drawn i.i.d. from a probability distribution P . The distribution P is initially unknown and we learn it by interacting repeatedly with the environment.

Many real-world optimization problems can be formulated in our setting, such as building a spanning tree for network routing [20]. When the delays on the links of the network are stochastic and their distribution is known, this problem can be solved by finding a minimum spanning tree. When the distribution is unknown, it must be learned, perhaps by exploring routing networks that seem initially suboptimal. We return to this problem in our experiments.

This paper makes three main contributions. First, we bring together the concepts of matroids [25] and bandits [15, 3], and propose a new class of combinatorial bandits, *matroid bandits*. On one hand, matroid bandits can be viewed as a new learning framework for a broad and important class of combinatorial optimization problems. On the other hand, matroid bandits are a class of K -step bandit problems that can be solved both computationally and sample efficiently.

Second, we propose a simple greedy algorithm for solving our problem, which explores based on the optimism in the face of uncertainty. We refer to our approach as *Optimistic Matroid Maximization* (OMM). OMM is both computationally and sample efficient. In particular, the time complexity of OMM is $O(L \log L)$ per episode, comparable to that of sorting L numbers. Moreover, the expected cumulative regret of OMM is sublinear in the number of episodes, and at most linear in the number of items L and the maximum number of chosen items K .

Finally, we evaluate OMM on three real-world problems. In

the first problem, we learn routing networks. In the second problem, we learn a policy for assigning loans in a micro-finance network that maximizes chances that the loans are repaid. In the third problem, we learn a movie recommendation policy. All three problems can be solved efficiently in our framework. This demonstrates that OMM is practical and can solve a wide range of real-world problems.

We adopt the following notation. We write $A + e$ instead of $A \cup \{e\}$, and $A + B$ instead of $A \cup B$. We also write $A - e$ instead of $A \setminus \{e\}$, and $A - B$ instead of $A \setminus B$.

2 Matroids

A *matroid* is a pair $M = (E, \mathcal{I})$, where $E = \{1, \dots, L\}$ is a set of L items, called the *ground set*, and \mathcal{I} is a family of subsets of E , called the *independent sets*. The family \mathcal{I} is defined by the following properties. First, $\emptyset \in \mathcal{I}$. Second, every subset of an independent set is independent. Finally, for any $X \in \mathcal{I}$ and $Y \in \mathcal{I}$ such that $|X| = |Y| + 1$, there must exist an item $e \in X - Y$ such that $Y + e \in \mathcal{I}$. This is known as the *augmentation property*. We denote by:

$$E(X) = \{e : e \in E - X, X + e \in \mathcal{I}\} \quad (1)$$

the set of items that can be added to set X such that the set remains independent.

A set is a *basis* of a matroid if it is a maximal independent set. All bases of a matroid have the same cardinality [25], which is known as the *rank* of a matroid. In this work, we denote the rank by K .

A *weighted matroid* is a matroid associated with a vector of non-negative weights $\mathbf{w} \in (\mathbb{R}^+)^L$. The e -th entry of \mathbf{w} , $\mathbf{w}(e)$, is the weight of item e . We denote by:

$$f(A, \mathbf{w}) = \sum_{e \in A} \mathbf{w}(e) \quad (2)$$

the sum of the weights of all items in set A . The problem of finding a *maximum-weight basis* of a matroid:

$$A^* = \arg \max_{A \in \mathcal{I}} f(A, \mathbf{w}) = \arg \max_{A \in \mathcal{I}} \sum_{e \in A} \mathbf{w}(e) \quad (3)$$

is a well-known combinatorial optimization problem. This problem can be solved greedily (Algorithm 1). The greedy algorithm has two main stages. First, A^* is initialized to \emptyset . Second, all items in the ground set are sorted according to their weights, from the highest to the lowest, and greedily added to A^* in this order. The item is added to the set A^* only if it does not make the set dependent.

3 Matroid Bandits

A *minimum spanning tree* is a maximum-weight basis of a matroid. The ground set E of this matroid are the edges of

Algorithm 1 The greedy method for finding a maximum-weight basis of a matroid [8].

Input: Matroid $M = (E, \mathcal{I})$, weights \mathbf{w}

```

 $A^* \leftarrow \emptyset$ 
Let  $e_1, \dots, e_L$  be an ordering of items such that:
 $\mathbf{w}(e_1) \geq \dots \geq \mathbf{w}(e_L)$ 
for all  $i = 1, \dots, L$  do
  if ( $e_i \in E(A^*)$ ) then
     $A^* \leftarrow A^* + e_i$ 
  end if
end for

```

a graph. A set of edges is considered to be independent if it does not contain a cycle. Each edge e is associated with a weight $\mathbf{w}(e) = u_{\max} - \mathbf{u}(e)$, where $u_{\max} = \max_e \mathbf{u}(e)$ and $\mathbf{u}(e)$ is the weight of edge e in the original graph.

The minimum spanning tree cannot be computed when the weights $\mathbf{w}(e)$ of the edges are unknown. This may happen in practice. For instance, consider the problem of building a routing network, which is represented as a spanning tree, where the expected delays on the links of the network are initially unknown. In this work, we study a variant of maximizing a modular function on a matroid that can address this kind of problems.

3.1 Model

We formalize our learning problem as a matroid bandit. A *matroid bandit* is a pair (M, P) , where M is a matroid and P is a probability distribution over the weights $\mathbf{w} \in \mathbb{R}^L$ of items E in M . The e -th entry of \mathbf{w} , $\mathbf{w}(e)$, is the weight of item e . The weights \mathbf{w} are stochastic and drawn i.i.d. from the distribution P . We denote the expected weights of the items by $\bar{\mathbf{w}} = \mathbb{E}[\mathbf{w}]$ and assume that each of these weights is non-negative, $\bar{\mathbf{w}}(e) \geq 0$ for all $e \in E$.

Each item e is associated with an *arm* and we assume that *multiple arms* can be pulled. A subset of arms $A \subseteq E$ can be pulled if and only if A is an independent set. The return for pulling arms A is $f(A, \mathbf{w})$ (Equation 2), the sum of the weights of all items in A . After the arms A are pulled, we observe the weight of each item in A , $\mathbf{w}(e)$ for all $e \in A$. This model of feedback is known as *semi-bandit* [2].

We assume that the matroid M is known and that the distribution P is unknown. Without loss of generality, we assume that the support of P is a bounded subset of $[0, 1]^L$. We would like to stress that we do not make any structural assumptions on P .

The optimal solution to our problem is a maximum-weight basis in expectation:

$$A^* = \arg \max_{A \in \mathcal{I}} \mathbb{E}_{\mathbf{w}}[f(A, \mathbf{w})] = \arg \max_{A \in \mathcal{I}} \sum_{e \in A} \bar{\mathbf{w}}(e). \quad (4)$$

Algorithm 2 OMM: Optimistic matroid maximization.

Input: Matroid $M = (E, \mathcal{I})$

// Initialization

Observe $\mathbf{w}_0 \sim P$

$\hat{w}_{e,1} \leftarrow \mathbf{w}_0(e) \quad \forall e \in E$

$T_e(0) \leftarrow 1 \quad \forall e \in E$

for all $t = 1, \dots, n$ **do**

 // Compute UCBs

$U_t(e) \leftarrow \hat{w}_{e,T_e(t-1)} + c_{t-1,T_e(t-1)} \quad \forall e \in E$

 // Find a maximum-weight basis with respect to U_t

$A^t \leftarrow \emptyset$

 Let e_1^t, \dots, e_L^t be an ordering of items such that:

$U_t(e_1^t) \geq \dots \geq U_t(e_L^t)$

for all $i = 1, \dots, L$ **do**

if ($e_i^t \in E(A^t)$) **then**

$A^t \leftarrow A^t + e_i^t$

end if

end for

 Observe $\{\mathbf{w}_t(e) : e \in A^t\}$, where $\mathbf{w}_t \sim P$

 // Update statistics

$T_e(t) \leftarrow T_e(t-1) \quad \forall e \in E$

$T_e(t) \leftarrow T_e(t) + 1 \quad \forall e \in A^t$

$\hat{w}_{e,T_e(t)} \leftarrow \frac{T_e(t-1)\hat{w}_{e,T_e(t-1)} + \mathbf{w}_t(e)}{T_e(t)} \quad \forall e \in A^t$

end for

The above optimization problem is equivalent to the problem in Equation 3. Therefore, it can be solved greedily by Algorithm 1 when the expected weights $\bar{\mathbf{w}}$ are known.

Our learning problem is *episodic*. In episode t , we choose a basis A^t and gain $f(A^t, \mathbf{w}_t)$, where \mathbf{w}_t is the realization of the stochastic weights in episode t . Our goal is to learn a policy, a sequence of bases, that minimizes the *expected cumulative regret* in n episodes:

$$R(n) = \mathbb{E}_{\mathbf{w}_1, \dots, \mathbf{w}_n} \left[\sum_{t=1}^n R_t(\mathbf{w}_t) \right], \quad (5)$$

where $R_t(\mathbf{w}_t) = f(A^*, \mathbf{w}_t) - f(A^t, \mathbf{w}_t)$ is the regret in episode t .

3.2 Algorithm

Our solution is designed based on the *optimism in the face of uncertainty* principle [17]. In particular, it is a variant of the greedy method for finding a maximum-weight basis of a matroid where the expected weight $\bar{\mathbf{w}}(e)$ of each item e is substituted with its optimistic estimate $U_t(e)$. Therefore, we refer to our approach as *Optimistic Matroid Maximization* (OMM).

The pseudocode of our algorithm is given in Algorithm 2. The algorithm can be summarized as follows. First, at the beginning of each episode t , we compute the *upper confidence bound (UCB)* on the weight of each item e :

$$U_t(e) = \hat{w}_{e,T_e(t-1)} + c_{t-1,T_e(t-1)}, \quad (6)$$

where $\hat{w}_{e,T_e(t-1)}$ is our estimate of $\bar{\mathbf{w}}(e)$ at the beginning of episode t , $c_{t-1,T_e(t-1)}$ represents the radius of the confidence interval around this estimate, and $T_e(t-1)$ is the number of times that OMM chooses item e before episode t . Second, we order all items e by their UCBs (Equation 6), from the highest to the lowest, and then add them greedily to A^t in this order. The item is added to the set A^t only if it does not make the set dependent. Finally, we choose the basis A^t , observe the weights of all items in the basis, and update our model $\hat{\mathbf{w}}$ of the world.

The radius:

$$c_{t,s} = \sqrt{2 \log(t)/s} \quad (7)$$

is defined such that each upper confidence bound $U_t(e)$ is with high probability an upper bound on the weight $\bar{\mathbf{w}}(e)$. The role of the UCBs is to encourage exploration of items that are not chosen very often. As the number of episodes increases, the estimates of the weights $\bar{\mathbf{w}}$ improve and OMM starts exploiting best items. The $\log(t)$ term increases with time t and enforces exploration, to avoid linear regret.

OMM is a greedy algorithm and therefore is extremely computationally efficient. In particular, let the time complexity of checking for independence, $e_i^t \in E(A^t)$, be $O(g(|A^t|))$. Then the time complexity of OMM is $O(L(\log L + g(K)))$ per episode, comparable to that of sorting L numbers. The design of our algorithm is not surprising and is motivated by prior work [12]. The main challenge is to derive a tight upper bound on the regret of OMM, which would reflect the structure of our problem.

4 Analysis

In this section, we analyze the regret of OMM. Our analysis is organized as follows. First, we introduce basic concepts and notation. Second, we show how to decompose the regret of OMM in a single episode. In particular, we partition the regret of a suboptimal basis into the sum of the regrets of individual items. This part of the analysis relies heavily on the structure of a matroid and is the most novel. Third, we derive two upper bounds, gap-dependent and gap-free, on the regret of OMM. Fourth, we prove a lower bound that matches the gap-dependent upper bound. Finally, we summarize the results of our analysis.

4.1 Notation

Before we present our results, we introduce notation used in our analysis. The *optimal basis* is $A^* = \{a_1^*, \dots, a_K^*\}$.

We assume that the items in A^* are ordered such that a_k^* is the k -th item with the highest expected weight. In episode t , OMM chooses a basis $A^t = \{a_1^t, \dots, a_K^t\}$, where a_k^t is the k -th item chosen by OMM. We say that item e is *suboptimal* if it belongs to $\bar{A}^* = E - A^*$, the *set of suboptimal items*. For any pair of suboptimal and optimal items, $e \in \bar{A}^*$ and a_k^* , we define a *gap*:

$$\Delta_{e,k} = \bar{\mathbf{w}}(a_k^*) - \bar{\mathbf{w}}(e) \quad (8)$$

and use it as a measure of how difficult it is to discriminate the items. For every item $e \in \bar{A}^*$, we define a set:

$$\mathcal{O}_e = \{k : \Delta_{e,k} > 0\}, \quad (9)$$

the indices of items in A^* whose expected weight is higher than that of item e . The cardinality of \mathcal{O}_e is $K_e = |\mathcal{O}_e|$.

4.2 Regret Decomposition

Our decomposition is motivated by the observation that all bases of a matroid are of the same cardinality. As a result, the difference in the expected values of any two bases can be always written as the sum of differences in the weights of their items. In particular:

$$\mathbb{E}_{\mathbf{w}}[f(A^*, \mathbf{w}) - f(A^t, \mathbf{w})] = \sum_{k=1}^K \Delta_{a_k^t, \pi(k)}, \quad (10)$$

where $\pi : \{1, \dots, K\} \rightarrow \{1, \dots, K\}$ is an arbitrary bijection from A^t to A^* such that $\pi(k)$ is the index of the item in A^* that is paired with the k -th item in A^t . In this work, we focus on one particular bijection.

Lemma 1. *For any two matroid bases A^* and A^t , there exists a bijection $\pi : \{1, \dots, K\} \rightarrow \{1, \dots, K\}$ such that:*

$$\{a_1^t, \dots, a_{k-1}^t, a_{\pi(k)}^*\} \in \mathcal{I} \quad \forall k = 1, \dots, K.$$

In addition, $\pi(k) = i$ when $a_k^t = a_i^$ for some i .*

Proof. The lemma is proved in Appendix. ■

The bijection π in Lemma 1 has two important properties. First, $\{a_1^t, \dots, a_{k-1}^t, a_{\pi(k)}^*\} \in \mathcal{I}$ for all k . In other words, OMM can choose item $a_{\pi(k)}^*$ at step k . However, OMM selects item a_k^t . By the design of OMM, this can happen only when the UCB of item a_k^t is larger or equal to that of item $a_{\pi(k)}^*$. As a result, we know that $U_t(a_k^t) \geq U_t(a_{\pi(k)}^*)$ in all steps k . Second, Lemma 1 guarantees that every optimal item in A^t is paired with the same item in A^* .

In the rest of the paper, we represent the bijection π using an indicator function. The indicator function:

$$\mathbb{1}_{e,k}(t) = \mathbb{1}\{\exists i : a_i^t = e, \pi(i) = k\} \quad (11)$$

indicates the event that item e is chosen instead of item a_k^* in episode t . Based on our new representation, we rewrite Equation 10 as:

$$\begin{aligned} \sum_{k=1}^K \Delta_{a_k^t, \pi(k)} &= \sum_{e \in \bar{A}^*} \sum_{k=1}^K \Delta_{e,k} \mathbb{1}_{e,k}(t) \\ &\leq \sum_{e \in \bar{A}^*} \sum_{k=1}^{K_e} \Delta_{e,k} \mathbb{1}_{e,k}(t) \end{aligned} \quad (12)$$

and then bound it from above. The last inequality is due to neglecting the negative gaps.

The above analysis applies to any basis A^t in any episode t . The results of our analysis are summarized below.

Theorem 1. *The expected regret of choosing any basis A^t in episode t is bounded as:*

$$\mathbb{E}_{\mathbf{w}}[f(A^*, \mathbf{w}) - f(A^t, \mathbf{w})] \leq \sum_{e \in \bar{A}^*} \sum_{k=1}^{K_e} \Delta_{e,k} \mathbb{1}_{e,k}(t).$$

The indicator function $\mathbb{1}_{e,k}(t)$ indicates the event that item e is chosen instead of item a_k^ in episode t . When the event $\mathbb{1}_{e,k}(t)$ happens, $U_t(e) \geq U_t(a_k^*)$. Moreover:*

$$\begin{aligned} \sum_{e \in \bar{A}^*} \sum_{k=1}^{K_e} \mathbb{1}_{e,k}(t) &\leq K \quad \forall t \\ \sum_{k=1}^{K_e} \mathbb{1}_{e,k}(t) &\leq 1 \quad \forall t, e \in \bar{A}^*. \end{aligned}$$

The last two inequalities follow from the fact that $\mathbb{1}_{e,k}(t)$ is a bijection from A^t to A^* , every item in the suboptimal basis A^t is matched with one unique item in A^* .

One remarkable aspect of our regret decomposition is that the exact form of the bijection is not required in the rest of our analysis. We only rely on the properties of $\mathbb{1}_{e,k}(t)$ that are stated in Theorem 1.

4.3 Upper Bounds

Our first result is a gap-dependent bound.

Theorem 2 (gap-dependent bound). *The expected cumulative regret of OMM is bounded as:*

$$R(n) \leq \sum_{e \in \bar{A}^*} \frac{16}{\Delta_{e,K_e}} \log n + \sum_{e \in \bar{A}^*} \sum_{k=1}^{K_e} \Delta_{e,k} \frac{4}{3} \pi^2.$$

Proof. First, we bound the expected regret in episode t us-

ing Theorem 1:

$$\begin{aligned}
R(n) &= \sum_{t=1}^n \mathbb{E}_{\mathbf{w}_1, \dots, \mathbf{w}_{t-1}} [\mathbb{E}_{\mathbf{w}_t} [R_t(\mathbf{w}_t)]] \\
&\leq \sum_{t=1}^n \mathbb{E}_{\mathbf{w}_1, \dots, \mathbf{w}_{t-1}} \left[\sum_{e \in \bar{A}^*} \sum_{k=1}^{K_e} \Delta_{e,k} \mathbb{1}_{e,k}(t) \right] \\
&= \sum_{e \in \bar{A}^*} \sum_{k=1}^{K_e} \Delta_{e,k} \mathbb{E}_{\mathbf{w}_1, \dots, \mathbf{w}_n} \left[\sum_{t=1}^n \mathbb{1}_{e,k}(t) \right]. \quad (13)
\end{aligned}$$

Second, we bound the expected cumulative regret associated with each item $e \in \bar{A}^*$. The key idea of this step is to decompose the indicator $\mathbb{1}_{e,k}(t)$ as:

$$\begin{aligned}
\mathbb{1}_{e,k}(t) &= \mathbb{1}_{e,k}(t) \mathbb{1}\{T_e(t-1) \leq \ell_{e,k}\} + \\
&\quad \mathbb{1}_{e,k}(t) \mathbb{1}\{T_e(t-1) > \ell_{e,k}\} \quad (14)
\end{aligned}$$

and choose $\ell_{e,k}$ appropriately. By Lemma 2, the regret associated with $T_e(t-1) > \ell_{e,k}$ is bounded as:

$$\begin{aligned}
&\sum_{k=1}^{K_e} \Delta_{e,k} \mathbb{E}_{\mathbf{w}_1, \dots, \mathbf{w}_n} \left[\sum_{t=1}^n \mathbb{1}_{e,k}(t) \mathbb{1}\{T_e(t-1) > \ell_{e,k}\} \right] \\
&\leq \sum_{k=1}^{K_e} \Delta_{e,k} \frac{4}{3} \pi^2 \quad (15)
\end{aligned}$$

when $\ell_{e,k} = \left\lfloor \frac{8}{\Delta_{e,k}^2} \log n \right\rfloor$. For the same value of $\ell_{e,k}$, the regret associated with $T_e(t-1) \leq \ell_{e,k}$ is bounded as:

$$\begin{aligned}
&\sum_{k=1}^{K_e} \Delta_{e,k} \mathbb{E}_{\mathbf{w}_1, \dots, \mathbf{w}_n} \left[\sum_{t=1}^n \mathbb{1}_{e,k}(t) \mathbb{1}\{T_e(t-1) \leq \ell_{e,k}\} \right] \\
&\leq \max_{\mathbf{w}_1, \dots, \mathbf{w}_n} \left[\sum_{t=1}^n \sum_{k=1}^{K_e} \Delta_{e,k} \mathbb{1}_{e,k}(t) \times \right. \\
&\quad \left. \mathbb{1}\left\{T_e(t-1) \leq \frac{8}{\Delta_{e,k}^2} \log n\right\} \right]. \quad (16)
\end{aligned}$$

The next step of our proof is based on three observations. First, the gaps are ordered such that $\Delta_{e,1} \geq \dots \geq \Delta_{e,K_e}$. Second, by the design of OMM, the counter $T_e(t)$ increases when the event $\mathbb{1}_{e,k}(t)$ happens, for any k . Finally, by Theorem 1, $\sum_{k=1}^{K_e} \mathbb{1}_{e,k}(t) \leq 1$ for any given e and t . Based on these facts, we bound Equation 16 from above by:

$$\left[\Delta_{e,1} \frac{1}{\Delta_{e,1}^2} + \sum_{k=2}^{K_e} \Delta_{e,k} \left(\frac{1}{\Delta_{e,k}^2} - \frac{1}{\Delta_{e,k-1}^2} \right) \right] 8 \log n. \quad (17)$$

By Lemma 3, the above term is bounded by $\frac{16}{\Delta_{e,K_e}} \log n$. Finally, we combine all of the above inequalities and get:

$$\begin{aligned}
&\sum_{k=1}^{K_e} \Delta_{e,k} \mathbb{E}_{\mathbf{w}_1, \dots, \mathbf{w}_n} \left[\sum_{t=1}^n \mathbb{1}_{e,k}(t) \right] \\
&\leq \frac{16}{\Delta_{e,K_e}} \log n + \sum_{k=1}^{K_e} \Delta_{e,k} \frac{4}{3} \pi^2. \quad (18)
\end{aligned}$$

Our main claim is obtained by summing over all suboptimal items $e \in \bar{A}^*$. ■

We also prove a gap-free bound.

Theorem 3 (gap-free bound). *The expected cumulative regret of OMM is bounded as:*

$$R(n) \leq 8\sqrt{KLn \log n} + \frac{4}{3} \pi^2 KL.$$

Proof. The key idea is to decompose the expected cumulative regret of OMM into two parts, where the gaps are larger than ε and at most ε . We analyze each part separately and then set ε to get the desired result.

Let $K_{e,\varepsilon}$ be the number of optimal items whose expected weight is higher than that of item e by more than ε and:

$$Z_{e,k}(n) = \mathbb{E}_{\mathbf{w}_1, \dots, \mathbf{w}_n} \left[\sum_{t=1}^n \mathbb{1}_{e,k}(t) \right]. \quad (19)$$

Then, based on Equation 13, the regret of OMM is bounded for any ε as:

$$\begin{aligned}
R(n) &\leq \sum_{e \in \bar{A}^*} \sum_{k=1}^{K_{e,\varepsilon}} \Delta_{e,k} Z_{e,k}(n) + \\
&\quad \sum_{e \in \bar{A}^*} \sum_{k=K_{e,\varepsilon}+1}^{K_e} \Delta_{e,k} Z_{e,k}(n). \quad (20)
\end{aligned}$$

The first term can be bounded similarly to Equation 18:

$$\begin{aligned}
&\sum_{e \in \bar{A}^*} \sum_{k=1}^{K_{e,\varepsilon}} \Delta_{e,k} Z_{e,k}(n) \\
&\leq \sum_{e \in \bar{A}^*} \frac{16}{\Delta_{e,K_{e,\varepsilon}}} \log n + \sum_{e \in \bar{A}^*} \sum_{k=1}^{K_{e,\varepsilon}} \Delta_{e,k} \frac{4}{3} \pi^2 \\
&\leq \frac{16}{\varepsilon} L \log n + \frac{4}{3} \pi^2 KL. \quad (21)
\end{aligned}$$

The second term is bounded trivially as:

$$\sum_{e \in \bar{A}^*} \sum_{k=K_{e,\varepsilon}+1}^{K_e} \Delta_{e,k} Z_{e,k}(n) \leq \varepsilon Kn \quad (22)$$

because all gaps $\Delta_{e,k}$ are bounded by ε and the maximum number of suboptimally chosen items in n episodes is Kn (Theorem 1). Based on our upper bounds, we get:

$$R(n) \leq \frac{16}{\varepsilon} L \log n + \varepsilon Kn + \frac{4}{3} \pi^2 KL \quad (23)$$

and then set $\varepsilon = 4\sqrt{\frac{L \log n}{Kn}}$. This concludes our proof. ■

4.4 Lower Bounds

We derive an asymptotic lower bound on the expected cumulative regret $R(n)$ that has the same dependence on the gap and n as the upper bound in Theorem 2. This bound is proved on a class of matroid bandits that are equivalent to K Bernoulli bandits.

Specifically, we prove the lower bound on a *partition matroid bandit*, which is defined as follows. Let E be a set of L items and B_1, \dots, B_K be a partition of this set. Let the family of independent sets be defined as:

$$\mathcal{I} = \{I \subseteq E : (\forall k : |I \cap B_k| \leq 1)\}. \quad (24)$$

Then $M = (E, \mathcal{I})$ is a *partition matroid* of rank K . Let P be a probability distribution over the weights of the items, where the weight of each item is distributed independently of the other items. Let the weight of item e be drawn i.i.d. from a Bernoulli distribution with mean:

$$\bar{w}(e) = \begin{cases} 0.5 & e = \min_{i \in B_k} i \\ 0.5 - \Delta & \text{otherwise,} \end{cases} \quad (25)$$

where $\Delta > 0$. Then $\tilde{B} = (M, P)$ is our partition matroid bandit. The key property of \tilde{B} is that it is equivalent to K independent Bernoulli bandits, one for each partition. The optimal item in each partition is the item with the smallest index, $\min_{i \in B_k} i$. All gaps are Δ .

To formalize our result, we need to introduce the notion of *consistent algorithms*. We say that the algorithm is consistent if for any matroid bandit, any suboptimal $e \in \bar{A}^*$, and any $\alpha > 0$, $\mathbb{E}[T_e(n)] = o(n^\alpha)$, where $T_e(n)$ is the number of times that item e is chosen in n episodes. In the rest of our analysis, we focus only on consistent algorithms. This is without loss of generality. In particular, by definition, an inconsistent algorithm performs poorly on some problems, and therefore extremely well on others. Because of this, it is difficult to prove good problem-dependent lower bounds for inconsistent algorithms. Our main claim is below.

Theorem 4. *For any partition matroid bandit \tilde{B} that is defined in Equations 24 and 25, and parameterized by L , K , and $0 < \Delta < 0.5$; the regret of any consistent algorithm is bounded from below as:*

$$\liminf_{n \rightarrow \infty} \frac{R(n)}{\log n} \geq \frac{L - K}{4\Delta}.$$

Proof. The theorem is proved as follows:

$$\begin{aligned} \liminf_{n \rightarrow \infty} \frac{R(n)}{\log n} &\geq \sum_{k=1}^K \sum_{e \in B_k - A^*} \frac{\Delta}{\text{kl}(0.5 - \Delta, 0.5)} \\ &= \frac{(L - K)\Delta}{\text{kl}(0.5 - \Delta, 0.5)} \\ &\geq \frac{L - K}{4\Delta}, \end{aligned} \quad (26)$$

where $\text{kl}(0.5 - \Delta, 0.5)$ is the KL divergence between two Bernoulli variables with means $0.5 - \Delta$ and 0.5 . The first inequality is due to Theorem 2.2 [4], which is applied separately to each partition B_k . The second inequality is due to $\text{kl}(p, q) \leq \frac{(p-q)^2}{q(1-q)}$, where $p = 0.5 - \Delta$ and $q = 0.5$. ■

4.5 Summary of Theoretical Results

We prove two upper bounds on the regret of OMM, one gap-dependent and one gap-free. These bounds can be summarized as:

$$\begin{aligned} \text{Theorem 2} & \quad O(L(1/\Delta) \log n) \\ \text{Theorem 3} & \quad O(\sqrt{KLn \log n}), \end{aligned} \quad (27)$$

where $\Delta = \min_e \min_{k \in \mathcal{O}_e} \Delta_{e,k}$. Both bounds are sublinear in the number of episodes n , and at most linear in the rank K of the matroid and the number of items L . In other words, they scale favorably with all quantities of interest and as a result we expect them to be practical.

Our upper bounds are reasonably tight. More specifically, the gap-dependent upper bound in Theorem 2 matches the lower bound in Theorem 4, which is proved on a partition matroid bandit. Furthermore, the gap-free upper bound in Theorem 3 matches the lower bound of Audibert *et al.* [2] in adversarial combinatorial semi-bandits, up to a factor of $\sqrt{\log n}$.

Our gap-dependent upper bound has the same form as the bound of Auer *et al.* [3] for multi-armed bandits. This observation suggests that the sample complexity of learning a maximum-weight basis of a matroid is comparable to that of the multi-armed bandit. The only major difference is in the definitions of the gaps. We conclude that learning with matroids is extremely sample efficient.

5 Experiments

Our algorithm is evaluated on three matroid bandit problems: graphic (Section 5.1), transversal (Section 5.2), and linear (Section 5.3).

All experiments are episodic. In each episode, OMM selects a basis A^t , observes the weights of the individual items in that basis, and then updates its model of the environment. The performance of OMM is measured by the *expected per-step return* in n episodes:

$$\frac{1}{n} \mathbb{E}_{\mathbf{w}_1, \dots, \mathbf{w}_n} \left[\sum_{t=1}^n f(A^t, \mathbf{w}_t) \right], \quad (28)$$

the expected cumulative return in n episodes divided by n . OMM is compared to two baselines. The first baseline is the maximum-weight basis A^* in expectation. The basis A^* is computed as in Equation 4 and is our notion of optimality.

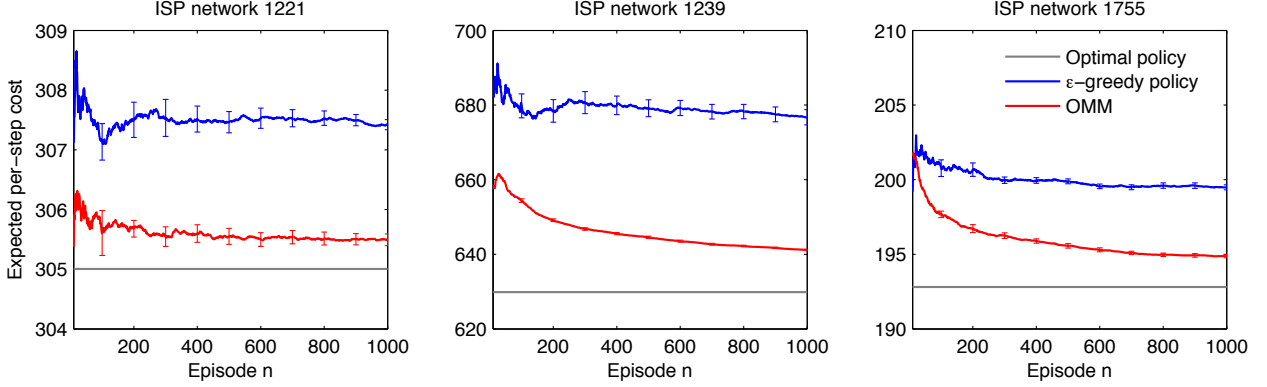


Figure 1: The expected per-step cost of building three minimum spanning trees in up to 10^3 episodes.

ISP network	Number of nodes	Number of edges	Minimum latency	Maximum latency	Average latency	Optimal policy	ϵ -greedy policy	OMM
1221	108	153	1	17	2.78	305.00	307.42 ± 0.08	305.49 ± 0.10
1239	315	972	1	64	3.20	629.88	676.74 ± 2.03	641.17 ± 0.18
1755	87	161	1	31	2.91	192.81	199.49 ± 0.16	194.88 ± 0.11
3257	161	328	1	47	4.30	550.85	570.35 ± 0.63	559.80 ± 0.10
3967	79	147	1	44	5.19	306.80	320.30 ± 0.52	308.54 ± 0.08
6461	141	374	1	45	6.32	376.27	424.78 ± 1.54	381.48 ± 0.07

Table 1: The description of six ISP networks from our experiments and the expected per-step cost of building minimum spanning trees on these networks in 10^3 episodes. All latencies and costs are in milliseconds.

The second baseline is an ϵ -greedy policy, where $\epsilon = 0.1$. This setting of ϵ is common in practice and corresponds to 10% exploration.

5.1 Graphic Matroid

In the first experiment, we evaluate OMM on the problem of learning a routing network for an Internet service provider (ISP). We make the assumption that the routing network is a spanning tree. Our objective is to learn a tree that has the lowest expected latency on its edges.

Our problem can be formulated as a *graphic matroid bandit*. The ground set E are the edges of a graph, which represents the topology of a network. We experiment with six networks from the *RocketFuel* dataset [23], which contain up to 300 nodes and 10^3 edges (Table 1). A set of edges is considered *independent* if it does not contain a cycle. The latency of edge e is $w(e) = \bar{w}(e) - 1 + \epsilon$, where $\bar{w}(e)$ is the expected latency, which is recorded in our dataset; and $\epsilon \sim \text{Exp}(1)$ is exponential noise. The latency $\bar{w}(e)$ ranges from one to 64 milliseconds. Our noise model is motivated by the following observation. The latency in ISP networks can be mostly explained by geographical distances [7], the expected latency $\bar{w}(e)$. The noise tends to be small, on the order of a few hundred microseconds, and it is unlikely to cause high latency.

In Figure 1, we report our results from three ISP networks.

We observe the same trends on all networks. First, the expected cost of OMM approaches that of the optimal solution A^* as the number of episodes increases. Second, OMM outperforms the ϵ -greedy policy in less than 10 episodes. The expected costs of all policies on all networks are reported in Table 1. We observe again that OMM consistently outperforms the ϵ -greedy policy, often by a large margin.

OMM learns quickly because all of our networks are sparse. In particular, the number of edges in each network is never more than four times larger than the number of edges in its spanning tree. Therefore, at least in theory, each edge can be observed at least once in four episodes and our method can learn quickly the mean latency of each edge.

5.2 Transversal Matroid

In the second experiment, we study the assignment of lending institutions (known as *partners*) to *lenders* in a micro-finance setting, such as Kiva [1]. This problem can be formulated under a family of matroids, called *transversal matroids* [9]. The ground set E of a transversal matroid is the set of left vertices of the corresponding bipartite graph, and the independence set \mathcal{I} consists of the sets of left vertices that belong to all possible matchings in the graph such that no two edges in a matching share an endpoint. The weight $\bar{w}(e)$ is the weight associated with the left vertices of the bipartite graph. The goal is to learn a transversal of the bipartite graph that maximizes the overall weight of selected

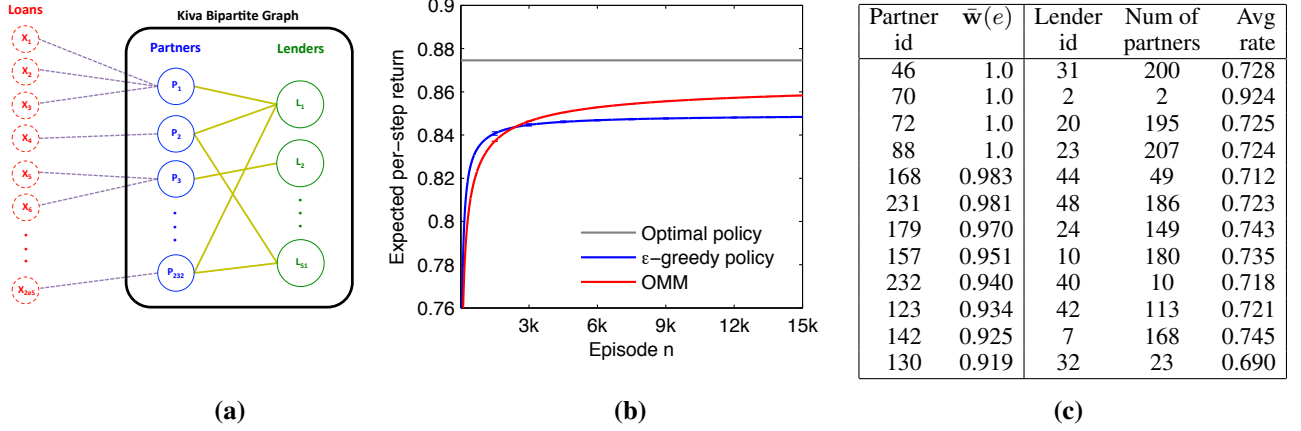


Figure 2: **(a)** The Kiva dataset can be modeled as a bipartite graph connecting lenders to field partners, which, in turn, fund several loans in the region. **(b)** The expected per-step return of finding maximum weight transversal in up to 15k episodes. **(c)** Top 12 selected partners assigned based on their mean success rate in the optimal solution A^* . The optimal solution involves 46 partner/lender assignments.

left vertices.

We used a sample of 194, 876 loans from the Kiva microfinance dataset [1], and created a bipartite graph. Every loan is handled by a partner (Figure 2-a). There are a total of 232 partners in the dataset that represent the left vertices of the bipartite graph and therefore the ground set E of the matroid. There are 286, 874 lenders in the dataset. We grouped these lenders into 51 clusters according to their location: 50 representing each individual state in United States, and 1 representing all foreign lenders. These 51 lender clusters constitute the right vertices of the bipartite graph. There is an edge between a partner and a lender if the lender is among the top 50% supporters of the partner, resulting in approximately 5k edges in the bipartite graph. The weight $\bar{w}(e)$ is the probability that a loan handled by partner e will be paid back. We estimate it from the dataset as $\bar{w}(e) = \frac{1}{n_l} \sum_{i=1}^{n_l} w_i(e)$, where n_l is the number of loans handled by this partner. We assume $w_i(e)$ is 0.7 if the loan i is in repayment, 1 if it is paid, and 0 otherwise. At the beginning of each episode, we choose the loan i at random.

The optimal solution A^* is a transversal in the graph that maximizes the overall success rate of the selected partners. Top twelve partners selected based on their mean success rate in the optimal solution are shown in Figure 2-c. For each partner, the id of the lender to which this partner was assigned along with the number of eligible partners of the lender and their average success rate are listed in the Table. The objective of OMM and ϵ -greedy policies is similar to the optimal policy with the difference that success rates (i.e. $w(e)$) are not known beforehand, and they must be learned by interacting repeatedly with the environment. Comparison results of the three policies are reported in Figure 2-b. Similar to the previous experiment, we observe the following trends. First, the expected return of OMM approaches

that of the optimal solution A^* as the number of episodes increases. Second, OMM outperforms the ϵ -greedy policy.

5.3 Linear Matroid

In the last experiment, we evaluate OMM on the problem of learning a set of diverse and popular movies. This kind of movies is typically recommended by existing content recommender systems. The movies are popular, and therefore the user is likely to choose them. The movies are diverse, and therefore cover many potential interests of the user.

Our problem can be formulated as a *linear matroid bandit*. The ground set E are movies from the *MovieLens* dataset [16], a dataset of 6 thousand people who rated one million movies. We restrict our attention to 25 most rated movies and 75 movies that are not well known. So the cardinality of E is 100. For each movie e , we define a feature vector $\mathbf{u}_e \in \{0, 1\}^{18}$, where $\mathbf{u}_e(j)$ indicates that movie e belongs to genre j . A set of movies X is considered *independent* if for any movie $e \in X$, the vector \mathbf{u}_e cannot be written as a linear combination of the feature vectors of the remaining movies in X . This is our notion of diversity. The expected weight $\bar{w}(e)$ is the probability that movie e is chosen. We estimate it as $\bar{w}(e) = \frac{1}{n_p} \sum_{i=1}^{n_p} w_i(e)$, where $w_i(e)$ is the indicator that person i rated movie e and n_p is the number of people in our dataset. At the beginning of each episode, we choose the person i at random.

Twelve most popular movies from the optimal solution A^* are listed in Figure 3. These movies cover a wide range of movie genres and appear to be diverse. This validates our assumption that linear independence is suitable for modeling diversity. The expected return of OMM is reported in the same figure. We observe the same trends as in the previous experiments. More specifically, the expected return of OMM

Movie title	$\bar{w}(e)$	Movie genres
American Beauty	0.568	Comedy Drama
Jurassic Park	0.442	Action Adventure Sci-Fi
Saving Private Ryan	0.439	Action Drama War
Matrix	0.429	Action Sci-Fi Thriller
Back to the Future	0.428	Comedy Sci-Fi
Silence of the Lambs	0.427	Drama Thriller
Men in Black	0.420	Action Adventure Comedy Sci-Fi
Fargo	0.416	Crime Drama Thriller
Shakespeare in Love	0.392	Comedy Romance
L.A. Confidential	0.379	Crime Film-Noir Mystery Thriller
E.T. the Extra-Terrestrial	0.376	Children's Drama Fantasy Sci-Fi
Ghostbusters	0.361	Comedy Horror

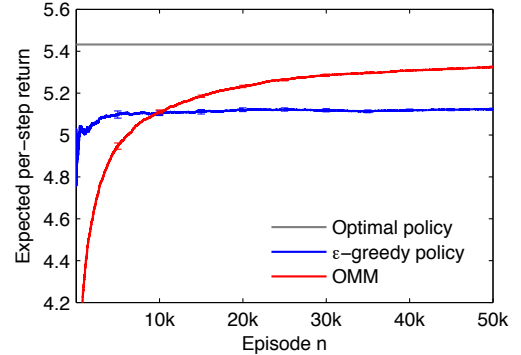


Figure 3: **Left.** Twelve most popular movies in the optimal solution A^* . The optimal solution involves 17 movies. **Right.** The expected per-step return of three movie recommendation policies in up to 50k episodes.

approaches that of A^* as the number of episodes increases and OMM outperforms the ϵ -greedy policy in 10k episodes.

6 Related Work

Our problem can be viewed as a stochastic combinatorial semi-bandit [12], where all feasible solutions are the independent sets of a matroid. Stochastic combinatorial semi-bandits were pioneered by Gai *et al.* [12], who proposed a UCB algorithm for solving these problems. Chen *et al.* [6] proved that the expected cumulative regret of this method is $O(K^2 L(1/\Delta) \log n)$. Our gap-dependent regret bound is $O(L(1/\Delta) \log n)$, a factor of K^2 tighter than the bound of Chen *et al.* [6]. Our analysis relies heavily on the properties of our problem and therefore we can derive a much tighter bound.

COMBAND [5], OSMD [2], and FPL [19] are algorithms for adversarial combinatorial semi-bandits. The main limitation of COMBAND and OSMD is that they are not guaranteed to be computationally efficient. More specifically, COMBAND needs to sample from a distribution over exponentially many solutions and OSMD needs to project to the convex hull of these solutions. FPL is computationally efficient but not very practical because its time complexity increases with time. On the other hand, OMM is guaranteed to be computationally efficient but can only solve a special class of combinatorial bandits, matroid bandits.

Matroids are a broad and important class of combinatorial optimization problems [21], which has been an active area of research for the past 80 years. This is the first paper that studies a well-known matroid problem in the bandit setting and proposes a learning algorithm for solving it.

Our work is also related to submodularity [18]. In particular, let:

$$g(X) = \max_{Y: Y \subseteq X, Y \in \mathcal{I}} f(Y, \bar{w}) \quad (29)$$

be the maximum weight of an independent set in X . Then

it is easy to show that $g(X)$ is submodular and monotonic in X , and that the maximum-weight basis of a matroid is a solution to $A^* = \arg \max_{A: |A|=K} g(A)$. Many algorithms for learning how to maximize a submodular function have been proposed recently [13, 26, 10, 24, 11]. None of these algorithms are suitable for solving our problem. There are two reasons. First, each algorithm is designed to maximize a specific submodular function and our function g may not be of that type. Second, the algorithms are only near optimal, learn a set A such that $g(A) \geq (1 - 1/e)g(A^*)$. Note that our method is guaranteed to be optimal and learn A^* .

7 Conclusions

This is the first work that studies the problem of learning a maximum-weight basis of a matroid, where the weights of the items are initially unknown, and have to be learned by interacting repeatedly with the environment. We propose a practical algorithm for solving this problem and bound its regret. The regret is sublinear in time and at most linear in all other quantities of interest. We evaluate our method on three real-world problems and show that it is practical.

Our regret bounds are $\Omega(\sqrt{L})$. Therefore, OMM is not practical when the number of items L is large. We believe that these kinds of problems can be solved efficiently by introducing additional structure, such as *linear generalization*. In this case, the weight of each item would be modeled as a linear function of its features and the goal is to learn the parameters of this function.

Many combinatorial optimization problems can be viewed as optimization on a matroid or its generalizations, such as *maximum-weight matching* on a bipartite graph and *minimum cost flows*. In a sense, these are the hardest problems in combinatorial optimization that can be solved optimally in polynomial time [22]. In this work, we show that one of these problems is efficiently learnable. We believe that the key ideas in our work are quite general and can be applied to other problems that involve matroids.

References

- [1] KIVA. <http://build.kiva.org/docs/data>, 2013.
- [2] Jean-Yves Audibert, Sebastien Bubeck, and Gabor Lugosi. Regret in online combinatorial optimization. *Mathematics of Operations Research*, 39(1):31–45, 2014.
- [3] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256, 2002.
- [4] Sébastien Bubeck and Nicolò Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 2012.
- [5] Nicolò Cesa-Bianchi and Gábor Lugosi. Combinatorial bandits. *Journal of Computer and System Sciences*, 78(5):1404–1422, 2012.
- [6] Wei Chen, Yajun Wang, and Yang Yuan. Combinatorial multi-armed bandit: General framework and applications. In *Proceedings of the 30th International Conference on Machine Learning*, pages 151–159, 2013.
- [7] Baek-Young Choi, Sue Moon, Zhi-Li Zhang, Konstantina Papagiannaki, and Christophe Diot. Analysis of point-to-point packet delay in an operational network. In *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies*, 2004.
- [8] Jack Edmonds. Matroids and the greedy algorithm. *Mathematical Programming*, 1(1):127–136, 1971.
- [9] Jack Edmonds and Delbert Ray Fulkerson. Transversals and matroid partition. *Journal of Research of the National Bureau of Standards*, 69B(3):147–153, 1965.
- [10] Victor Gabillon, Branislav Kveton, Zheng Wen, Brian Eriksson, and S. Muthukrishnan. Adaptive submodular maximization in bandit setting. In *Advances in Neural Information Processing Systems 26*, pages 2697–2705, 2013.
- [11] Victor Gabillon, Branislav Kveton, Zheng Wen, Brian Eriksson, and S. Muthukrishnan. Large-scale optimistic adaptive submodularity. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, 2014.
- [12] Yi Gai, Bhaskar Krishnamachari, and Rahul Jain. Combinatorial network optimization with unknown variables: Multi-armed bandits with linear rewards and individual observations. *IEEE/ACM Transactions on Networking*, 20(5):1466–1478, 2012.
- [13] Andrew Guillory and Jeff Bilmes. Online submodular set cover, ranking, and repeated active learning. In *Advances in Neural Information Processing Systems 24*, pages 1107–1115, 2011.
- [14] Naoki Katoh. Combinatorial optimization algorithms in resource allocation problems. *Encyclopedia of Optimization*, pages 259–264, 2001.
- [15] T. L. Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6(1):4–22, 1985.
- [16] Shyong Lam and Jon Herlocker. MovieLens 1M Dataset. <http://www.grouplens.org/node/12>, 2012.
- [17] Rémi Munos. The optimistic principle applied to games, optimization, and planning: Towards foundations of Monte-Carlo tree search. *Foundations and Trends in Machine Learning*, 2012.
- [18] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions - I. *Mathematical Programming*, 14(1):265–294, 1978.
- [19] Gergely Neu and Gábor Bartók. An efficient algorithm for learning with semi-bandit feedback. In *Proceedings of the 24th International Conference on Algorithmic Learning Theory*, pages 234–248, 2013.
- [20] Carlos Oliveira and Panos Pardalos. A survey of combinatorial optimization problems in multicast routing. *Computers and Operations Research*, 32(8):1953–1981, 2005.
- [21] James Oxley. *Matroid Theory*. Oxford University Press, New York, NY, 2011.
- [22] Christos Papadimitriou and Kenneth Steiglitz. *Combinatorial Optimization*. Dover Publications, Mineola, NY, 1998.
- [23] Neil Spring, Ratul Mahajan, and David Wetherall. Measuring ISP topologies with Rocketfuel. *IEEE / ACM Transactions on Networking*, 12(1):2–16, 2004.
- [24] Zheng Wen, Branislav Kveton, Brian Eriksson, and Sandilya Bhamidipati. Sequential Bayesian search. In *Proceedings of the 30th International Conference on Machine Learning*, pages 977–983, 2013.
- [25] Hassler Whitney. On the abstract properties of linear dependence. *American Journal of Mathematics*, 57(3):509–533, 1935.
- [26] Yisong Yue and Carlos Guestrin. Linear submodular bandits and their application to diversified retrieval. In *Advances in Neural Information Processing Systems 24*, pages 2483–2491, 2011.

Multi-label Image Classification with A Probabilistic Label Enhancement Model

Xin Li and Feipeng Zhao and Yuhong Guo
Department of Computer and Information Sciences
Temple University
Philadelphia, PA 19122, USA
{xinli, feipeng.zhao, yuhong}@temple.edu

Abstract

In this paper, we present a novel probabilistic label enhancement model to tackle multi-label image classification problem. Recognizing multiple objects in images is a challenging problem due to label sparsity, appearance variations of the objects and occlusions. We propose to tackle these difficulties from a novel perspective by constructing auxiliary labels in the output space. Our idea is to exploit label combinations to enrich the label space and improve the label identification capacity in the original label space. In particular, we identify a set of informative label combination pairs by constructing a tree-structured graph in the label space using the maximum spanning tree algorithm, which naturally forms a conditional random field. We then use the produced label pairs as auxiliary new labels to augment the original labels and perform piecewise training under the framework of conditional random fields. In the test phase, max-product message passing is used to perform efficient inference on the tree graph, which integrates the augmented label pair classifiers and the standard individual binary classifiers for multi-label prediction. We evaluate the proposed approach on several image classification datasets. The experimental results demonstrate the superiority of our label enhancement model in terms of both prediction performance and running time comparing to the-state-of-the-art multi-label learning methods.

1 INTRODUCTION

With the development of internet and digital devices, the availability of visual data has been dramatically increasing in recent decades, which provides billions of images and videos. An important task for information retrieval and processing over such image and video data is object-

based image annotation, which requires identifying a set of objects presented in each image from a given set of desired object concepts. The image annotation problem for object recognition is an inherent multi-label classification problem, since each image usually contains more than one object of interest. Multi-label classification generalizes the standard multi-class classification by allowing each instance to be simultaneously assigned into multiple label categories. A key challenge for multi-label classification is label sparsity. That is, the multiple labels are supported by the training data at different levels and many rare labels may lack sufficient training supports to be reliably recognized individually. Hence instead of learning binary classifiers independently for each label, many multi-label learning methods have proposed to exploit label correlations or label dependence to improve multi-label classification performance, including second-order strategy methods [7, 11, 15], which model pairwise label correlations, and high-order strategy methods [16, 19, 36], which consider the interactions among subsets of labels.

Moreover, on image classification, multi-label learning also faces the general intra-class variation challenge of standard multi-class classification which is caused by viewpoint and context variations and occlusions, as shown in Figure 1. From the figures, we can see that the appearance of the object *people* can be profoundly different across different images, by co-occurring with different objects and having different occlusion patterns. In such cases, an individual binary classifier may not be reliable for recognizing a target object. But other co-occurred objects can likely provide some useful information. For example, in the image “*people riding a bike*”, many parts of the *bike* are invisible, but the *people* is easy to recognize and can provide information about the *bike*; in the image “*people sitting in the car*”, each *people* is severely occluded but the *car* can be detected easily and help the recognition of the *people* if they often co-occur; in the image “*people riding a horse*”, the objects *people* and *horse* can be helpful to each other as well. Moreover, the recognition of such composite co-occurrence patterns can also help the correct recognition of individual objects, especially for the ones that are occluded



Figure 1: Examples of image occlusion and object co-occurrence patterns in object recognition tasks.

or difficult to be recognized individually.

Motivated by these observations, in this paper, we propose a novel probabilistic label enhancement model that utilizes the label combination patterns to improve multi-label image classification performance. Our assumption is that visual composites can be helpful when single classifier fails. For example, assume the composite “*people riding horses*” often stays in similar poses. A classifier for this visual composite can then capture the co-appearance of the two objects even though both *people* and *horse* classifiers fail to reliably recognize them separately. We propose to construct label combinations, in particular label pairs, as auxiliary new labels to augment the original labels and improve label identification capacity in the original label space. In particular, we identify a set of informative label combination pairs by constructing a tree-structured graph in the label space using the maximum spanning tree algorithm, according to the label co-occurrence information in the training data. This naturally forms a conditional random field framework, under which we perform piecewise training. The label pairs identified by the edges of the tree are used as auxiliary new labels, and a binary classifier is trained for each label in the augmented label space. To integrate the augmented multiple binary classifiers for multi-label prediction in the original label space, we perform exact inference on the tree-structured conditional random field using max-product message passing. We evaluate the proposed approach on a number of multi-label image classification datasets. The experimental results show that the proposed label enhancement model effectively outperforms the related state-of-the-art methods in terms of both prediction performance and running time.

The rest of the paper is organized as follows. In Section 2, we present a brief review over the related work. The proposed approach is presented in Section 3. We report the experimental results in Section 4 and finally conclude the paper in Section 5.

2 RELATED WORK

A considerable amount of research has been devoted to addressing image annotation and multi-label classification

problems in the literature. In this section, we will provide a brief review over the most related work to the proposed approach from the perspectives of image annotation, object interaction and multi-label classification.

Image Annotation There are three major groups of image annotation techniques [13]: (i) Generative models. Some methods in this group use generative topic models such as latent Dirichlet allocation [1], probabilistic latent semantic analysis [24], and hierarchical Dirichlet processes [33]. They model annotated images as samples from a mixture of topics, where each topic is a distribution over image features. Some other methods use mixture models to define a joint distribution over image features and annotation tags [4, 9, 22]. However, generative models perform training by maximizing generative data likelihoods, which are not necessarily optimal for the target prediction performance. (ii) Discriminative models. The methods in this group address image annotation as a classification problem. For example, simple methods in [23, 12] treat labels independently and learn a classifier for each label, while more advanced methods in [30, 3] improve the classification performance by considering the co-occurrences of different labels. (iii) Nearest neighbor based models. For example, the label propagation method in [13] constructs a similarity graph for all images, and propagates the label information via the graph; and the search based method in [10] exploits a regression based kernel metric.

Object Interaction There are a number of works on object interaction that share the same intuition as our proposed work, that is, visual composites can be helpful while single components fail. The work in [18] learns object interactions by modeling the prepositions and adjectives that relate nouns. The work in [34] models the co-occurrence of objects and human poses in human-object interaction activities. In [8], the interactions between objects are modeled implicitly in the context of predicting sentences for images. [28] introduces a complex visual composite concept called visual phrase for object detection, which treats each phrase as a new label. Though this work shares similarity with our proposed work in exploiting visual composites, there are significant differences between it and our work. First, its visual phrases are not automatically discovered but predefined. By contrast, the label combinations in our work are

constructed automatically. Second, it addresses very different problems from ours. It tackles object detection tasks while we address multi-label image annotation problems; its goal is to find a bounding box where the visual composite occurs, while our goal is to predict the category labels of an image.

Multi-label Classification The most straightforward multi-label classification method is binary relevance [2], which trains a binary classifier for each label. The obvious flaw of such method is the complete ignorance of label correlations. Hence, numerous methods that encode label correlations have been proposed. One group is the ranking based methods [7, 11, 32], which rank the relevant labels higher than irrelevant ones and capture label correlations implicitly in the loss function. This technique however relies on a good distance metric and a fine-tuned threshold in determining the number of relevant labels. The method in [15] hence further eliminates this drawback by developing a novel calibrated separation ranking loss function. Another group is the graph-based methods, which implicitly incorporate label correlations into label propagation algorithms as either part of the graph weights [20, 5] or additional constraints [30, 35]. There are also a set of probabilistic graph-based methods [6, 14, 17, 27]. The method in [14] uses directed graphs over the label variables to capture label dependence under a probabilistic conditional dependency network model. [17] further improves this model by learning sparse conditional dependency graphs. [6, 27] integrate multiple classifiers in a chain graph to capture label correlations. These methods share similarity with our proposed approach in capturing label correlations by integrating probabilistic classifiers on graphs over labels. However, [6, 27] are limited to chain graphs and they apply greedy heuristics to search for the best label vector on each test instance; [14, 17] use cyclic directed graphs and their test phases involve approximate inference. By contrast, our method can exploit any automatically generated acyclic tree graphs, not necessarily chains, while using a max-product message passing algorithm to perform efficient exact inference.

3 PROPOSED MODEL

3.1 Preliminaries

Multi-label Classification systems can be described as below. Given the input feature space $\mathcal{X} \in \mathbb{R}^d$ and the output label space $\mathcal{Y} = \{0, 1\}^L$, a mapping function $\mathbf{h}: \mathcal{X} \rightarrow \mathcal{Y}$ can be used to predict the corresponding label vector $\mathbf{y} \in \mathcal{Y}$ for each input data instance $\mathbf{x} \in \mathcal{X}$. Multi-label learning focuses on identifying a good mapping function \mathbf{h} from the training data. The most straightforward method to learn such a mapping function is the binary relevance method, which assumes labels are independently generated and learns one binary classifier h_i for each label. The out-

put of \mathbf{h} is a L -length binary vector

$$\mathbf{h}(\mathbf{x}) = [h_1(\mathbf{x}), h_2(\mathbf{x}), \dots, h_L(\mathbf{x})].$$

The binary classifier h_i can be trained by minimizing different loss functions, such as log-loss and hinge loss.

Conditional Random Fields (CRFs) are undirected graphical models that model the conditional distribution of the output labels given an input vector based on undirected graphs. An undirected graph $G = (V, E)$ in the label space is formed by a set of vertices V , each of which represents a label variable, and a set of undirected edges E , where each edge consists of a pair of vertices $(s, t) \in E$ and represents the dependence relationship between the label variables. The joint probability of a configuration of the label variables in a CRF can be given by

$$P(\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_L | \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{c \in \mathcal{C}} \psi_c(\mathbf{y}_c, \mathbf{x})$$

where $Z(\mathbf{x})$ is a partition function that ensures a valid conditional distribution given the input data instance \mathbf{x} , \mathcal{C} is the set of cliques of the graph, and ψ_c is the potential function for clique c , which maps the clique label configuration \mathbf{y}_c and the input data instance \mathbf{x} into a positive scalar value. The standard training procedure of conditional random fields typically involves first-order gradient descent or second-order Newton methods, which require performing inference over each training instance in each iterative parameter update step. For general graphs, performing exact inference in CRFs is intractable, and approximate inference algorithms are usually used instead. Moreover, performing inference in each parameter update step can make the training process computationally expensive, especially for large and densely connected graphs and large training sets.

3.2 Probabilistic Label Enhancement Model

The straightforward method for multi-label image classification casts the problem as a set of independent binary classification problems, one for each object label, and trains one binary classifier for each label using the one-vs-all scheme. Training binary classifiers is computationally efficient and the one-vs-all training scheme can scale linearly with the increasing of the label set. However, as we discussed before, such binary classifiers can fail to accurately recognize the individual objects in an image, due to label sparsity, intra-class variations, and occlusions. In this work, we propose to enhance these standard binary classifiers by exploiting the label combination patterns which typically present as co-occurred object composites in images of the training data, as shown in Figure 1. We first identify the informative label combination pairs by learning a tree-structured undirected graph in the label space, which forms the structure of a conditional random field.

Then we use the label combination pairs as augmenting new labels and formulate the learning process as a piecewise training procedure under the framework of conditional random fields. Finally we apply the trained probabilistic model to predict labels for test images using a max-product exact inference algorithm.

3.2.1 Learning Tree-Structured Graph

Given the labeled training images $D = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^n$, where each label vector $\mathbf{y}^{(i)}$ contains $\{0,1\}$ values with length L , corresponding to the L label classes, we aim to identify the useful object composites by finding the informative label co-occurrence patterns. Though in principle we can consider any label combination patterns, for computational simplicity we focus on second-order patterns, i.e., label pairs. We take all possible label pairs as candidates by constructing a fully connected graph over the L label variables. Then we measure the combination strength of each label pair as the weight of the corresponding edge using an appropriate criterion. One standard criterion is the empirical *mutual information* measure, which is popularly used to measure the dependence strength of two variables and can be computed from the training data. For example, the empirical mutual information between label variables Y_i and Y_j can be computed as

$$MI(Y_i; Y_j) = \sum_{y_i, y_j \in \{0,1\}} \hat{P}(y_i, y_j) \log \left(\frac{\hat{P}(y_i, y_j)}{\hat{P}(y_i) \hat{P}(y_j)} \right)$$

with the empirical probabilities computed from the training data. However, this measure treats the co-presence of the two labels and the co-missing of them equivalently, while we want to find the label composites that have significant co-presence patterns. Hence we propose a simple new measure, *normalized co-occurrence*, to use. For two label variables Y_i and Y_j , the normalized co-occurrence measure is defined as

$$NC(Y_i; Y_j) = \frac{\text{count}(Y_i, Y_j)}{\min(\text{count}(Y_i), \text{count}(Y_j))}$$

where $\text{count}(Y_i, Y_j)$ is the number of co-occurrence of the two labels in the training data, such that

$$\text{count}(Y_i, Y_j) = \sum_{\ell=1}^n I[\mathbf{y}_i^{(\ell)} = 1, \mathbf{y}_j^{(\ell)} = 1] \quad (1)$$

and $I[\cdot]$ denotes an indicator function. Similarly, $\text{count}(Y_i)$ and $\text{count}(Y_j)$ are the numbers of occurrences of single labels in the training data. By normalizing the co-occurrence counts of the two labels with the minimum of their individual occurrence counts, the measure emphasizes the relative relatedness of the two objects and favors the less frequently appeared objects. For example, assume there are 15 images containing *people* and *dogs*, and 10 images containing *people* and *cars*, while there are totally 100 people images, 80

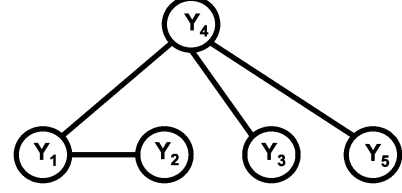


Figure 2: An example of the constructed tree-structured graph over labels.

dog images and 20 car images. The composite of *people* and *cars* can be more important to capture than the composite of *people* and *dogs*, towards the goal of assisting the objects with sparse supports in the training data. Our proposed measure encodes this principle.

Given the proposed normalized co-occurrence criterion, we can compute the weights for all edges between the label variables. Then we use a maximum spanning tree algorithm to select $(L - 1)$ edges according to the computed weights to form a tree-structured graph. In our implementation, we used Prim's algorithm [26] to produce the maximum spanning tree. Figure 2 demonstrates an example of the constructed tree graph over the label variables. The label pair connected by each edge on the constructed tree graph will be used as a constructed new label to augment the original labels. For example, for the tree graph in Figure 2, since there is an edge between the node Y_1 and the node Y_2 , we will consider a constructed new label $Y_{1 \sim 2}$, which has binary values $\{0, 1\}$. The label value for $Y_{1 \sim 2}$ in each instance can be produced based on that $Y_{1 \sim 2} = 1$ is equivalent to $Y_1 = 1 \wedge Y_2 = 1$. Then for each instance $\mathbf{x}^{(i)}$, we set $\mathbf{y}_{1 \sim 2}^{(i)} = 1$ if and only if the instance has been assigned both label Y_1 and label Y_2 such that $\mathbf{y}_1^{(i)} = 1$ and $\mathbf{y}_2^{(i)} = 1$. Otherwise, we have $\mathbf{y}_{1 \sim 2}^{(i)} = 0$. Thus each constructed new label can be treated as a new prediction class from the prediction perspective. The reason that we produce tree graphs instead of densely connected cyclic graphs is that tree graphs have acyclic structures and permit efficient exact inference in the test phase to integrate the augmented label classifiers with the binary classifiers in the original label space.

3.2.2 Piecewise Training of CRFs

The tree-structured graph constructed in the label space actually forms a standard CRF model that permits label vector prediction from the input data, where we treat each node and each edge as separate cliques. Based on our motivation of capturing object composite concept to help the multi-label prediction in the original label space, we propose to perform piecewise training for the tree-structured CRF by learning the potential functions for each node clique and each edge clique separately. That is, we train a set of L binary classifiers independently from the data, one for each

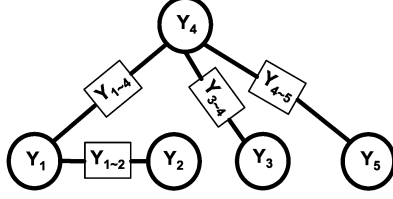


Figure 3: The factor graph constructed from the tree graph in Figure 2. The circle nodes are variable nodes and the rectangle nodes are factor nodes.

label in the original label space, as the potential functions for the node cliques, and train a set of $(L - 1)$ binary classifiers independently from the data for the constructed new labels as the potential functions for the corresponding edge cliques. Piecewise training can effectively avoid the repeated inference required for each step of parameter updates in the standard CRF training procedure and make the learning process efficient and scalable. It has been shown in [29] that piecewise training of a CRF can be justified as minimizing a family of upper bounds on the log partition function of the data log-likelihood.

To have the outputs of potential functions compatible to each other, we propose to use binary probabilistic classifiers, in particular binary logistic regression classifiers, for training. Each binary logistic regression classifier can be trained efficiently by using second-order Newton methods to minimize the regularized log-likelihood. For the k -th classifier, this is to minimize

$$\min_{\mathbf{w}} \sum_{i=1}^n \log \left(1 + e^{-\hat{\mathbf{y}}_k^{(i)} \mathbf{w}^\top \mathbf{x}^{(i)}} \right) + \frac{\beta}{2} \mathbf{w}^\top \mathbf{w} \quad (2)$$

where β is a trade-off parameter, and $\hat{\mathbf{y}}_k^{(i)}$ is simply the translation of $\mathbf{y}_k^{(i)}$ from values $\{1, 0\}$ to $\{1, -1\}$.

3.2.3 Inference with Max-product Algorithm

Given the trained tree-structured CRF model, the multi-label prediction on a test instance can be performed using the max-product inference algorithm [21]. The max-product algorithm conducts label decoding through message passing which operates in factor graphs. Given the trained pairwise CRF model, we then first transfer it into a factor graph by simply keeping all variable nodes and adding a factor node for each edge clique. For example, the factor graph constructed for the tree-structured CRF in Figure 2 is given in Figure 3, where each variable node is represented as a circle and each factor node is represented as a rectangle. For a given test instance \mathbf{x} , the potentials of the two types of nodes in the factor graph can be computed using the probabilistic binary classifiers produced in the training phase, such as $\psi(\mathbf{y}_i) = P(\mathbf{y}_i|\mathbf{x})$ and $\psi(\mathbf{y}_i, \mathbf{y}_j) = \psi(\mathbf{y}_{i \sim j}) = P(\mathbf{y}_{i \sim j}|\mathbf{x})$.

The decoding process on the test instance \mathbf{x} aims to find the maximum a posteriori (MAP) label assignment by solving

$$\mathbf{y}^* = \arg \max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}) \quad (3)$$

The max-product algorithm performs this decoding on the factor graph using the following message passing. First, we randomly select a variable node r as the root of the tree, and pass messages from leaves until they reach the root. There are two types of messages: node-to-factor messages and factor-to-node messages. The message from the node i to the factor a (e.g., $a = i \sim j$) can be computed as

$$\mu_{i \rightarrow a}(\mathbf{y}_i) = \psi(\mathbf{y}_i) \prod_{c \in N(i) \setminus a} \mu_{c \rightarrow i}(\mathbf{y}_i) \quad (4)$$

where $N(i) \setminus a$ represents all the neighboring factor nodes of node i excluding factor node a . The message from the factor $a = i \sim j$ to the node j can be computed as

$$\mu_{a \rightarrow j}(\mathbf{y}_j) = \max_{\mathbf{y}_i} \left(\psi_a(\mathbf{y}_i, \mathbf{y}_j) \mu_{i \rightarrow a}(\mathbf{y}_i) \right) \quad (5)$$

Back pointers are kept for each value that achieves the maximum at a max operation. At the root, we multiply all incoming messages to obtain the maximum probability and the MAP configuration of the root node \mathbf{y}_r^*

$$P^* = \max_{\mathbf{y}_r} \left(\psi(\mathbf{y}_r) \prod_{a \in N(r)} \mu_{a \rightarrow r}(\mathbf{y}_r) \right) \quad (6)$$

$$\mathbf{y}_r^* = \arg \max_{\mathbf{y}_r} \left(\psi(\mathbf{y}_r) \prod_{a \in N(r)} \mu_{a \rightarrow r}(\mathbf{y}_r) \right) \quad (7)$$

We then back trace the pointers and find the complete values \mathbf{y}^* that lead to P^* . For tree graphs, this max-product algorithm provides an exact inference solution. But for an arbitrary graph with loops, it can only provide an approximate solution.

4 EXPERIMENTS

To evaluate the proposed approach, we conducted experiments on several standard multi-label image classification datasets, comparing to a few state-of-the-art multi-label learning methods and baselines. We report our empirical results in this section.

4.1 Datasets

We used the following three image datasets in our experiments: *Pascal VOC 2007 (Pascal07)*, *Corel5K*, and *SUN 2012*. *Pascal07* is one of the most famous image datasets for classification and detection and it contains 20 different object classes. *Corel5K* is a standard image set for multi-label classification with 5,000 instances and 260 classes. To have a fair comparison with a few comparison methods,

some of which are too slow to deal with many classes, we selected two subsets with 50 most frequent labels and 100 most frequent labels respectively to use. *SUN 2012* [31] is a recently released large-scale image set for object detection. Similarly, we used two subsets with 50 labels and 100 labels respectively. The properties of the datasets used in our experiment are briefly summarized in Table 1, where cardinality denotes the average number of labels assigned to one image. In these datasets, each image is represented as a 512-dimension *GIST* [25] feature vector.

Table 1: Summary information of the datasets.

Dataset	#images	#labels	cardinality
Pascal07	4168	20	2.26
Corel5K(s50)	4999	50	2.32
Corel5K(s100)	4999	100	2.89
SUN12(s50)	5000	50	8.98
SUN12(s100)	5000	100	11.18

4.2 Experimental Results

On each of five datasets, we compared the proposed approach to the following state-of-the-art multi-label classification methods and baseline method:

- *Ensembled Probabilistic Classifier Chain (EPCC)*. This probabilistic multi-label learning method is developed in [6] and it integrates base classifiers in a chain structure in the label space.
- *Maximum Margin Output Coding (MMOC)*. This is a multi-label learning method developed in [36], which performs classification on a simultaneously learned lower-dimensional label space within a maximum margin framework.
- *Logistic Regression (LR)*. This is a baseline method that trains a set of independent binary logistic regression classifiers, one for each label, to perform multi-label classification.

For our proposed approach, there is one regularization trade-off parameter β to set for the logistic regression classifiers. We found that logistic regression classifiers are not very sensitive to this parameter. In our experiments, we set β as a very small value around 0.0002. For the comparison methods *EPCC* and *MMOC*, we used the code packages released on the internet¹. These packages contain parameter selection procedures and settings.

On each dataset, we performed a 5-fold cross validation to compare all the methods. To evaluate the multi-label classification results from different perspective, we used five

standard criteria: macro-F1, micro-F1, hamming loss, precision and recall. The average results and standard deviations in terms of the five criteria for all the four methods are reported in Table 2. We can see that our proposed method outperforms all the other comparison methods across all five datasets and in terms of all the four measure criteria: macro-F1, micro-F1, Precision and Recall. In terms of hamming loss, the proposed approach produced the best results on four out of the five datasets. Moreover, the proposed approach significantly outperforms the baseline *LR* method across all different settings. This clearly shows that the augmenting new labels in our model are very effective in assisting identifying the individual labels, and it is very beneficial to exploit the label co-occurrence patterns. By comparing the results on the *Corel5k(s50)* dataset and the *Corel5k(s100)* dataset, we can see that with the increasing of the label set size, the performance of all methods in terms of the four measures, macro-F1, micro-F1, precision and recall, has the general trend of decreasing. In terms of the hamming loss, however, the results of all approaches are even better on *Corel5k(s100)* than on *Corel5k(s50)*. This seems very strange. But if we check the two datasets, we can see that though *Corel5k(s100)* contains 50 more labels than *Corel5k(s50)*, the difference between their label cardinality values is very small. This indicates that the labels are even more sparse in *Corel5k(s100)* than in *Corel5k(s50)*. By producing similar number of positive labels, the performance of each approach will automatically get better in terms of hamming loss, with the increasing of the label set size. This result suggests that hamming loss is not an appropriate criterion for multi-label classification when the label cardinality is small while the number of label classes is large. Similar results are observed across *SUN12(s50)* and *SUN12(s100)* as well. Another observation over the table is that *EPCC* produced the second best results in most cases. The *EPCC* method greatly outperforms the baseline *LR* almost on all the datasets and in terms of all criteria, except that on *SUN12(s100)* in terms of macro-F1 and on *SUN12(s50)* in terms of precision, where it produces similar results with *LR*. The *MMOC* method is much more time-consuming than other methods. On the two datasets with 100 labels, it fails to yield any result within reasonable period of running time. It has inferior performance comparing to the proposed approach and *EPCC* in most cases.

Running time To compare the empirical efficiency of the approaches, we have also recorded the training time and testing time of each approach on a 64-bit machine with 16GB memory and quad core intel i7 processors. The results of average running time are reported in Figure 4. We can see the baseline *LR* is the most efficient method in terms of both training and testing time, since it only needs to train a set of binary classifiers and perform classification independently for each label. Among the remaining three methods, our proposed approach is significantly more ef-

¹ <https://github.com/multi-label-classification/PCC>;
<http://www.cs.cmu.edu/yizhang1/files/ICML2012.Code.zip>

Table 2: The average results and standard deviations of all the comparison methods on the five datasets in terms of different evaluation criteria. On each dataset, the best result in each criterion across different methods is shown in bold font. '-' denotes the fact that the method fails to run on the corresponding dataset due to the large label size.

Measure	Methods	Datasets				
		Pascal07	Corel5k(s50)	Corel5k(s100)	SUN12(s50)	SUN12(s100)
Macro-F1	Proposed	0.268±0.005	0.276±0.003	0.167±0.004	0.377±0.004	0.315±0.002
	EPCC	0.252±0.005	0.245±0.006	0.158±0.002	0.355±0.002	0.210±0.003
	MMOC	0.220±0.003	0.218±0.002	-	0.318±0.002	-
	LR	0.247±0.006	0.201±0.003	0.135±0.002	0.323±0.002	0.215±0.002
Micro-F1	Proposed	0.579±0.004	0.362±0.005	0.333±0.003	0.581±0.003	0.514±0.002
	EPCC	0.567±0.003	0.351±0.002	0.327±0.002	0.563±0.001	0.507±0.002
	MMOC	0.543±0.006	0.238±0.003	-	0.514±0.002	-
	LR	0.481±0.007	0.222±0.003	0.197±0.003	0.486±0.003	0.386±0.003
Hamming Loss	Proposed	0.057±0.003	0.062±0.002	0.023±0.002	0.146±0.003	0.089±0.004
	EPCC	0.094±0.001	0.077±0.001	0.047±0.000	0.166±0.001	0.110±0.000
	MMOC	0.089±0.002	0.057±0.001	-	0.154±0.001	-
	LR	0.121±0.002	0.079±0.001	0.049±0.000	0.170±0.001	0.138±0.001
Precision	Proposed	0.697±0.013	0.343±0.003	0.310±0.005	0.656±0.003	0.541±0.003
	EPCC	0.649±0.004	0.311±0.002	0.300±0.003	0.544±0.001	0.517±0.002
	MMOC	0.689±0.011	0.225±0.004	-	0.614±0.003	-
	LR	0.518±0.010	0.198±0.003	0.185±0.004	0.548±0.004	0.403±0.003
Recall	Proposed	0.570±0.010	0.458±0.014	0.418±0.008	0.641±0.002	0.565±0.005
	EPCC	0.557±0.003	0.453±0.005	0.404±0.004	0.610±0.003	0.523±0.003
	MMOC	0.482±0.004	0.184±0.005	-	0.464±0.003	-
	LR	0.507±0.006	0.235±0.004	0.206±0.003	0.456±0.002	0.396±0.002

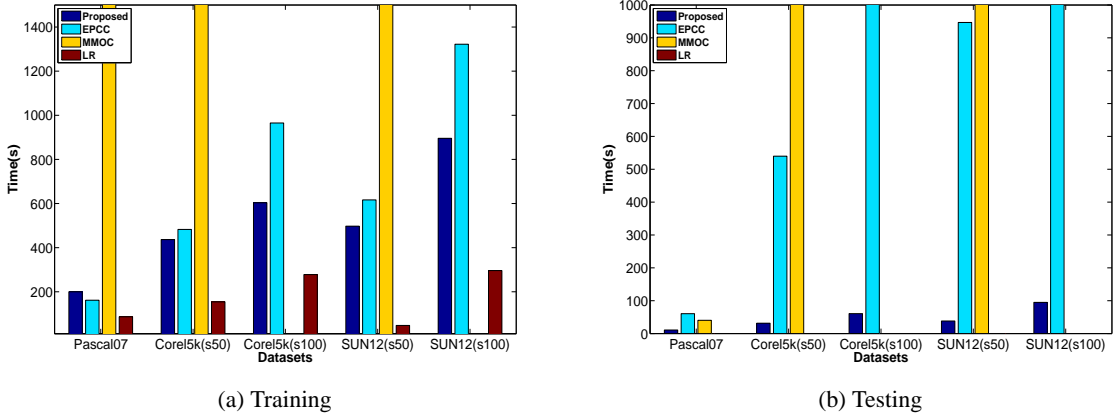






Figure 4: Training and testing time (seconds) for all methods. Note on Corel5k(s100) and SUN12(s100), the yellow bar (for MMOC) is missing due to that MMOC fails to handle these datasets.

ficient than the other two methods in terms of the testing time. For training time, the proposed approach is similar to *EPCC* on the dataset *Pascal07* which has small label set, and is more efficient than *EPCC* on the other larger scale datasets. *MMOC* is the most inefficient one among all the four methods. It even fails to produce any results on the two datasets with 100 labels.

Illustration of the Results To have an illustrative understanding about the image annotation problem and the prediction results, in Table 3 we presented the predicted labels on four testing images from the *SUN12(s50)* dataset by the four methods. The true positive labels are shown in bold font. We can see that our proposed approach is in general more accurate than the other methods, though *EPCC* has

Table 3: The predicted labels on four test images of SUN12(s50) by the comparison methods. The true positive labels are shown in bold font.

Methods				
Proposed	wall, floor, ceiling, chair, door, cabinet, table, vase, bottle, window	floor, wall, ceiling, door, table, person, box, books, chair	wall, door, road, car, sky, trees, person, mountain	door, sky, trees, grass
EPCC	wall, floor, ceiling, chair, ceiling lamp, table, vase, flowers, window, plant	wall, floor, ceiling, chair, door, person, ceiling lamp, window, cabinet	sky, window, door, plant, building, tree, grass	sky, tree, wall, floor, window, ceiling, chair, door, table, plant
MMOC	wall, floor, window, ceiling, chair, table, curtain, sofa, window	wall, floor, ceiling, person, window, ceiling lamp	sky, car, ceiling, grass, plant, building, tree, streetlight	sky, tree, wall, window, plants
LR	wall, floor, ceiling, chair, table, bottle, window, curtain, rug, sofa	wall, floor, ceiling, person	wall, sky, road, car, plant, building, tree, grass, streetlight	sky, tree, grass, plant, wall

good precision result on the first image as well.

All these results suggest that by capturing the object combination patterns in newly created labels, the proposed probabilistic label enhancement model provides an effective and efficient framework for multi-label image classification.

4.3 Experiments with Dense Graphs

Our proposed approach constructs a tree-graph to identify the informative label combination patterns. In order to produce the tree structure, the maximum spanning tree algorithm needs to ignore the edges with larger (normalized co-occurrence) weights to avoid cycles. To investigate whether this is problematic, we tried an alternative version of the proposed approach by constructing a densely connected graph for label combinations, instead of restricting to singly connected trees. Specifically, we produce the dense graph by simply keeping a proportion of the existing edges (there is no edge between label pairs that never co-occur) with largest weights. In the experiments, we kept the top 30% of the edges.

We compared the two variants of the proposed model across the five datasets. In particular, Figure 5 shows the examples of the dense graph and the tree graph constructed on the Pascal07 dataset. The tree graph only has 19 edges, while the dense graph has 37 edges. We can see that the two graphs have many common edges but also capture some very different label combination patterns. There are

some interesting pairs missing in the tree graph. For example, *sofa* and *tv monitor* can be observed in most sitting rooms, but their combination pair is not kept in the tree graph. On the other hand, the tree graph captured many important co-occurrence patterns with *much less* number of edges. For example, the tree graph captures the frequent co-occurrences between *person* and many other objects. By looking at the images in Pascal07, we can find many images containing *person* in different classes, which explains why the tree graph is *person*-centric. Moreover, even with much more edges, there are two isolated nodes in the dense graph, while none of nodes can be isolated in the tree graph.

The classification results of these two variants are reported in Table 4, in terms of macro-F1, micro-F1 and hamming loss. We can see that though the tree graph sacrificed edges with larger weights to maintain a singly connected tree structure, its performance is similar or even slightly better than the performance of the dense graph in most cases. In terms of the three measures, the dense graph only outperforms the tree graph on *Corel5k(s50)* and *SUN12(s50)* in terms of macro-F1, and on *Pascal07* in terms of hamming loss. With cyclic dense graphs, the max-product algorithm in the test phase can only produce approximate inference results. This can contribute to the inferior performance of the dense graph in many cases. Moreover, with more edges kept in the graph, there will be more auxiliary classifiers to train in the training phase, this makes the dense graph variant to have larger training time. All these information suggests that the tree graph is a desirable structure.

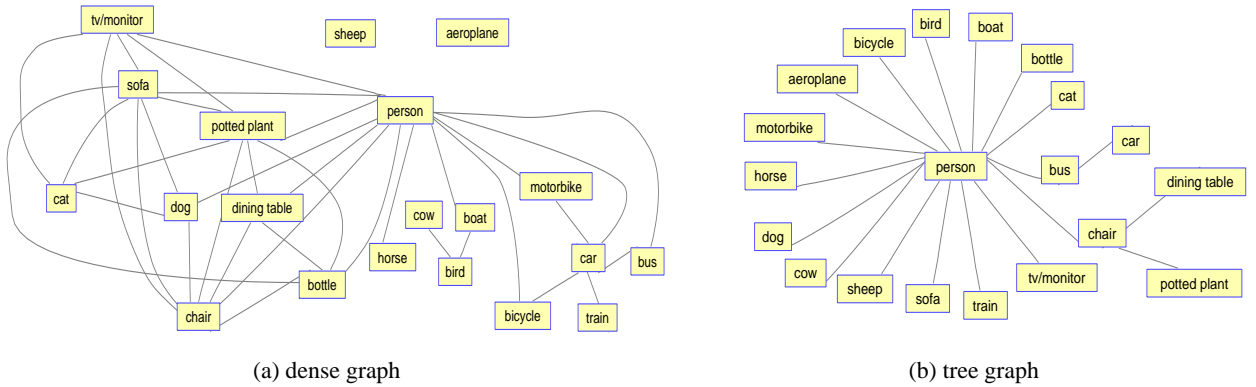


Figure 5: The densely connected graph and tree graph constructed on Pascal07.

Table 4: The average comparison results between the proposed approach (with tree graph) and its alternative version with a dense graph.

Measure	Methods	Datasets				
		Pascal07	Corel5k(s50)	Corel5k(s100)	SUN12(s50)	SUN12(s100)
Macro-F1	Tree	0.268±0.005	0.276±0.003	0.167±0.004	0.377±0.004	0.315±0.002
	Dense	0.250±0.005	0.278±0.005	0.162±0.003	0.420±0.005	0.298±0.002
Micro-F1	Tree	0.579±0.004	0.362±0.005	0.333±0.003	0.581±0.003	0.514±0.002
	Dense	0.568±0.002	0.360±0.004	0.330±0.003	0.579±0.003	0.504±0.003
Hamming Loss	Tree	0.057±0.003	0.062±0.002	0.023±0.002	0.146±0.003	0.089±0.004
	Dense	0.052±0.001	0.078±0.001	0.032±0.001	0.172±0.001	0.120±0.001

5 CONCLUSION

In this paper, we presented a novel probabilistic label enhancement model for multi-label image classification. The idea is to use informative label combination pairs (i.e., the object composite concepts in images) to augment the original labels which can be difficult to predict individually due to label sparsity, intra-class variations and occlusions, aiming to enhance the overall multi-label prediction performance. We formulated our model under the conditional random field framework by first constructing a tree graph in the label space based on the label co-occurrence patterns in the training data, and then performing efficient piecewise training. The learning process of the proposed model only requires training a set of independent binary classifiers, while its tree structure permits efficient and exact max-product inference in the test phase. Our experiments on several image classification datasets showed the proposed approach has superior performance in terms of both prediction quality and empirical computational complexity, comparing to the state-of-the-art comparison methods.

References

- [1] K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D. Blei, and M. Jordan. Matching words and pictures. *J. Mach. Learn. Res.*, 3:1107–1135, 2003.
- [2] M.R. Boutell, J. Luo, X. Shen, and C.M. Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757 – 1771, 2004.
- [3] X. Cai, F. Nie, W. Cai, and H. Huang. New graph structured sparsity model for multi-label image annotations. In *Proceedings of ICCV*, 2013.
- [4] G. Carneiro, A. Chan, P. Moreno, and N. Vasconcelos. Supervised learning of semantic classes for image annotation and retrieval. *IEEE TPAMI*, 29(3), 2007.
- [5] G. Chen, Y. Song, F. Wang, and C. Zhang. Semi-supervised multi-label learning by solving a sylvester equation. In *Proceedings of SDM*, 2008.
- [6] K. Dembczynski, W. Cheng, and E. Hüllermeier. Bayes optimal multilabel classification via probability classifier chains. In *Proceedings of ICML*, 2010.
- [7] A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. In *Proceedings of NIPS*, 2002.
- [8] A. Farhadi, M. Hejrati, M. Sadeghi, P. Young, C. Rashtchian, J. Hockenmaier, and D. Forsyth. Every picture tells a story: Generating sentences from images. In *Proceedings of ECCV*, 2010.

- [9] S. Feng, R. Manmatha, and V. Lavrenko. Multiple bernoulli relevance models for image and video annotation. In *Proceedings of CVPR*, 2004.
- [10] Z. Feng, R. Jin, and A. Jain. Large-scale image annotation by efficient and robust kernel metric learning. In *Proceedings of ICCV*, 2013.
- [11] J. Fürnkranz, E. Hüllermeier, E. Loza Mencía, and K. Brinker. Multilabel classification via calibrated label ranking. *Mach. Learn.*, 73(2):133–153, 2008.
- [12] D. Grangier and S. Bengio. A discriminative kernel-based approach to rank images from text queries. *IEEE TPAMI*, 30(8):1371–1384, 2008.
- [13] M. Guillaumin, T. Mensink, J. Verbeek, and C. Schmid. Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation. In *Proceedings of CVPR*, 2009.
- [14] Y. Guo and S. Gu. Multi-label classification using conditional dependency networks. In *Proceedings of IJCAI*, 2011.
- [15] Y. Guo and D. Schuurmans. Adaptive large margin training for multilabel classification. In *Proceedings of AAAI*, 2011.
- [16] Y. Guo and D. Schuurmans. Multi-label classification with output kernels. In *Proceedings of ECML*, 2013.
- [17] Y. Guo and W. Xue. Probabilistic multi-label classification with sparse feature learning. In *Proceedings of IJCAI*, 2013.
- [18] A. Gupta and L. Davis. Beyond nouns: Exploiting prepositions and comparative adjectives for learning visual classifiers. In *Proceedings of ECCV*, 2008.
- [19] S. Ji, L. Sun, R. Jin, and J. Ye. Multi-label multiple kernel learning. In *Proceedings of NIPS*, 2008.
- [20] F. Kang, R. Jin, and R. Sukthankar. Correlated label propagation with application to multi-label learning. In *Proceedings of CVPR*, 2006.
- [21] F. Kschischang, B. Frey, and H.A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. Inform. Theory*, 47:498–519, 2001.
- [22] V. Lavrenko, R. Manmatha, and J. Jeon. A model for learning the semantics of pictures. In *Proceedings in NIPS*, 2003.
- [23] T. Mensink, J. Verbeek, and G. Csurka. Learning structured prediction models for interactive image labeling. In *Proceedings of CVPR*, 2011.
- [24] F. Monay and D. Gatica-Perez. PLSA-based image auto-annotation: Constraining the latent space. In *Proceedings of MM*, 2004.
- [25] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001.
- [26] R.C. Prim. Shortest connection networks and some generalizations. *Bell System Technical Journal*, 36(6):1389–1401, 1957.
- [27] J. Read, B. Pfahringer, G. Holmes, and E. Frank. Classifier chains for multi-label classification. *Machine Learning and Knowledge Discovery in Databases*, 5782:254–269, 2009.
- [28] M. Sadeghi and A. Farhadi. Recognition using visual phrases. In *Proceedings of CVPR*, 2011.
- [29] C. Sutton and A. McCallum. Piecewise training of undirected models. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, 2005.
- [30] H. Wang, H. Huang, and C. Ding. Image annotation using multi-label correlated green’s function. In *Proceedings of CVPR*, 2009.
- [31] J. Xiao, J. Hays, K. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *Proceedings of CVPR*, 2010.
- [32] M. Xu, Y. Li, and Z. Zhou. Multi-label learning with pro loss. In *Proceedings of AAAI*, 2013.
- [33] O. Yakhnenko and V. Honavar. Annotating images and image objects using a hierarchical dirichlet process model. In *Proceedings of MDM*, 2008.
- [34] B. Yao and Fei-Fei Li. Modeling mutual context of object and human pose in human-object interaction activities. In *Proceedings of CVPR*, 2010.
- [35] Z. Zha, T. Mei, J. Wang, Z. Wang, and X. Hua. Graph-based semi-supervised learning with multiple labels. *J. Vis. Comun. Image Represent.*, 20(2):97–103, February 2009.
- [36] Y. Zhang and J. Schneider. Maximum margin output coding. In *Proceedings of ICML*, 2012.

Sequential Model-Based Ensemble Optimization

Alexandre Lacoste*

Alexandre.Lacoste.1@ulaval.ca
Département IFT-GLO
Université Laval
Québec, Canada

Hugo Larochelle

Hugo.Larochelle@usherbrooke.ca
Département d'informatique
Université de Sherbrooke
Québec, Canada

Mario Marchand

Mario.Marchand@ift.ulaval.ca
Département IFT-GLO
Université Laval
Québec, Canada

François Laviolette

Francois.Laviolette@ift.ulaval.ca
Département IFT-GLO
Université Laval
Québec, Canada

Abstract

One of the most tedious tasks in the application of machine learning is model selection, i.e. hyperparameter selection. Fortunately, recent progress has been made in the automation of this process, through the use of sequential model-based optimization (SMBO) methods. This can be used to optimize a cross-validation performance of a learning algorithm over the value of its hyperparameters. However, it is well known that ensembles of learned models almost consistently outperform a single model, even if properly selected. In this paper, we thus propose an extension of SMBO methods that automatically constructs such ensembles. This method builds on a recently proposed ensemble construction paradigm known as Agnostic Bayesian learning. In experiments on 22 regression and 39 classification data sets, we confirm the success of this proposed approach, which is able to outperform model selection with SMBO.

1 INTRODUCTION

The automation of hyperparameter selection is an important step towards making the practice of machine learning more approachable to the non-expert and increases its impact on data reliant sciences. Significant progress has been made recently, with many methods reporting success in tuning a large variety of algorithms [Bergstra et al. \[2011\]](#), [Hutter et al. \[2011\]](#), [Snoek et al. \[2012\]](#), [Thornton et al. \[2013\]](#). One successful general paradigm is known as Sequential Model-Based Optimization (SMBO). It is based on a process that alternates between the proposal of a new hyperparameter configuration to test and the update of an adaptive model of the relationship between hyperparameter configurations and their holdout set performances. Thus, as the model learns about this relationship, it increases its ability to suggest improved hyperparameter configurations

and gradually converges to the best solution.

While finding the single best model configuration is useful, better performance is often obtained by, instead, combining several (good) models into an ensemble. This was best illustrated by the winning entry of the Netflix competition, which combined a variety of models [\[Bell et al., 2007\]](#). Even if one concentrates on a single learning algorithm, combining models produced by using different hyperparameters is also helpful. Intuitively, models with comparable performances are still likely to generalize differently across the input space and produce different patterns of errors. By averaging their predictions, we can hope that the majority of models actually perform well on any given input and will move the ensemble towards better predictions globally, by dominating the average. In other words, the averaging of several comparable models reduces the variance of our predictor compared to each individual in the ensemble, while not sacrificing too much in terms of bias.

However, constructing such ensembles is just as tedious as performing model selection and at least as important in the successful deployment of machine-learning-based systems. Moreover, unlike the model selection case for which SMBO can be used, no comparable automatic ensemble construction methods have been developed thus far. The current methods of choice remain trial and error or exhaustive grid search for exploring the space of models to combine, followed by a selection or weighting strategy which is often an heuristic. One exception is the work of [Thornton et al. \[2013\]](#), which can support the construction of ensembles, but only of up to 5 models.

In this paper, we propose a method for leveraging the recent research on SMBO in order to generate an ensemble of models, as opposed to the single best model. The proposed approach builds on the Agnostic Bayes framework [\[Lacoste et al., 2014\]](#), which provides a successful strategy for weighting a predetermined and finite set of models (already trained) into an ensemble. Using a successful SMBO method, we show how we can effectively generalize this framework to the case of an infinite space of models (indexed by its hyperparameter space). The result-

ing method is simple and highly efficient. Our experiments on 22 regression and 39 classification data sets confirm that it outperforms the regular SMBO model selection method.

The paper develops as follows. First, we describe SMBO and its use for hyperparameter selection (Section 2). We follow with a description of the Agnostic Bayes framework and present a bootstrap-based implementation of it (Section 3). Then, we describe the proposed algorithm for automatically constructing an ensemble using SMBO (Section 4). Finally, related work is discussed (Section 5) and the experimental comparisons are presented (Section 6).

2 HYPERPARAMETER SELECTION WITH SMBO

Let us first lay down the notation we will be using to describe the task of model selection for a machine learning algorithm. In this setup, a task D corresponds to a probability distribution over the input-output space $\mathcal{X} \times \mathcal{Y}$. Given a set of examples $S \sim D^m$ (which will be our holdout validation set), the objective is to find, among a set \mathcal{H} , the *best* function $h^* : \mathcal{X} \rightarrow \mathcal{Y}$. In general, \mathcal{H} can be any set and we refer to a member as a predictor. In the context of hyperparameter selection, \mathcal{H} corresponds to the set of models trained on a training set $T \sim D^n$ (disjoint from S), for different configurations of the learning algorithm’s hyperparameters γ . Namely, let \mathcal{A}_γ be the learning algorithm with a hyperparameter configuration $\gamma \in \Gamma$, we will note $h_\gamma = \mathcal{A}_\gamma(T)$ the predictor obtained after training on T . The set \mathcal{H} contains all predictors obtained from each $\gamma \in \Gamma$ when \mathcal{A}_γ is trained on T , i.e. $\mathcal{H} \stackrel{\text{def}}{=} \{h_\gamma | \gamma \in \Gamma\}$.

To assess the quality of a predictor, we use a loss function

$$\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R},$$

that quantifies the penalty incurred when h_γ predicts $h_\gamma(x)$ while the true target is y . Then, we can define the risk $R_D(h_\gamma)$ as being the expected loss of h_γ on task D , i.e. $R_D(h_\gamma) \stackrel{\text{def}}{=} \mathbf{E}_{x,y \sim D} [\mathcal{L}(h_\gamma(x), y)]$. Finally, the *best*¹ function is simply the one minimizing the risk, i.e.

$$h^* \stackrel{\text{def}}{=} \underset{h_\gamma \in \mathcal{H}}{\operatorname{argmin}} R_D(h_\gamma).$$

Here, estimating h^* thus corresponds to hyperparameter selection.

For most of machine learning history, the state of the art in hyperparameter selection has been testing a list of predefined configurations and selecting the best according to the loss function \mathcal{L} on some holdout set of examples S . When a learning algorithm has more than one hyperparameter, a grid search is required, forcing $|\Gamma|$ to grow exponentially with the number of hyperparameters. In addition, the

¹The best solution may not be unique but any of them are equally good.

search may yield a suboptimal result when the minimum lies outside of the grid or when there is not enough computational power for an appropriate grid resolution. Recently, randomized search has been advocated as a better replacement to grid search [Bergstra and Bengio, 2012]. While it tends to be superior to grid search, it remains inefficient since its search is not informed by results of the sequence of hyperparameters that are tested.

To address these limitations, there has been an increasing amount of work on automatic hyperparameter optimization [Bergstra et al., 2011, Hutter et al., 2011, Snoek et al., 2012, Thornton et al., 2013]. Most rely on an approach called sequential model based optimization (SMBO). The idea consists in treating $R_S(h_\gamma) \stackrel{\text{def}}{=} f(\gamma)$ as a learnable function of γ , which we can learn from the observations $\{(\gamma_i, R_S(h_{\gamma_i}))\}$ collected during the hyperparameter selection process.

We must thus choose a model family for f . A common choice is a Gaussian process (GP) representation, which allows us to represent our uncertainty about f , i.e. our uncertainty about the value of $f(\gamma^*)$ at any unobserved hyperparameter configuration γ^* . This uncertainty can then be leveraged to determine an *acquisition function* that suggests the most promising hyperparameter configuration to test next.

Namely, let functions $\mu : \Gamma \rightarrow \mathbb{R}$ and $K : \Gamma \times \Gamma \rightarrow \mathbb{R}$ be the mean and covariance kernel functions of our GP over f . Let us also denote the set of the M previous evaluations as

$$\mathcal{R} \stackrel{\text{def}}{=} \{(\gamma_i, R_S(h_{\gamma_i}))\}_{i=1}^M \quad (1)$$

where $R_S(h_{\gamma_i})$ is the empirical risk of h_{γ_i} on set S , i.e. the holdout set error for hyperparameter γ .

The GP assumption on f implies that the conditional distribution $p(f(\gamma^*) | \mathcal{R})$ is Gaussian, that is

$$\begin{aligned} p(f(\gamma^*) | \mathcal{R}) &= \mathcal{N}(f(\gamma^*); \mu(\gamma^*; \mathcal{R}), \sigma^2(\gamma^*; \mathcal{R})), \\ \mu(\gamma^*; \mathcal{R}) &\stackrel{\text{def}}{=} \mu(\gamma^*) + \mathbf{k}^\top \mathbf{K}^{-1}(\mathbf{r} - \boldsymbol{\mu}), \\ \sigma^2(\gamma^*; \mathcal{R}) &\stackrel{\text{def}}{=} K(\gamma^*, \gamma^*) - \mathbf{k}^\top \mathbf{K}^{-1} \mathbf{k} \end{aligned}$$

where $\mathcal{N}(f(\gamma^*); \mu(\gamma^*; \mathcal{R}), \sigma^2(\gamma^*; \mathcal{R}))$ is the Gaussian density function with mean $\mu(\gamma^*; \mathcal{R})$ and variance $\sigma^2(\gamma^*; \mathcal{R})$. We also have vectors

$$\begin{aligned} \boldsymbol{\mu} &\stackrel{\text{def}}{=} [\mu(\gamma_1), \dots, \mu(\gamma_M)]^\top, \\ \mathbf{k} &\stackrel{\text{def}}{=} [K(\gamma^*, \gamma_1), \dots, K(\gamma^*, \gamma_M)]^\top, \\ \mathbf{r} &\stackrel{\text{def}}{=} [R_S(h_{\gamma_1}), \dots, R_S(h_{\gamma_M})]^\top, \end{aligned}$$

and matrix \mathbf{K} is such that $\mathbf{K}_{ij} = K(\gamma_i, \gamma_j)$.

There are several choices for the acquisition function. One that has been used with success is the one maximizing the *expected improvement*:

$$\text{EI}(\gamma^*; \mathcal{R}) \stackrel{\text{def}}{=} \mathbf{E}[\max\{r_{\text{best}} - f(\gamma^*), 0\} | \mathcal{R}] \quad (2)$$

which can be shown to be equal to

$$\sigma^2(\gamma^*; \mathcal{R}) (d(\gamma^*; \mathcal{R}) \Phi(d(\gamma^*; \mathcal{R})) + \mathcal{N}(d(\gamma^*; \mathcal{R}), 0, 1)) \quad (3)$$

where Φ is the cumulative distribution function of the standard normal and

$$r_{\text{best}} \stackrel{\text{def}}{=} \min_i R_S(h_{\gamma_i}),$$

$$d(\gamma^*; \mathcal{R}) \stackrel{\text{def}}{=} \frac{r_{\text{best}} - \mu(\gamma^*; \mathcal{R})}{\sigma(\gamma^*; \mathcal{R})}.$$

The acquisition function thus maximizes Equation 3 and returns its solution. This optimization can be performed by gradient ascent initialized at points distributed across the hyperparameter space according to a Sobol sequence, in order to maximize the chance of finding a global optima. One advantage of expected improvement is that it directly offers a solution to the exploration-exploitation trade-off that hyperparameter selection faces.

An iteration of SMBO requires fitting the GP to the current set of tested hyperparameters \mathcal{R} (initially empty), invoking the acquisition function, running the learning algorithm with the suggested hyperparameters and adding the result to \mathcal{R} . This procedure is expressed in Algorithm 1. Fitting the GP corresponds to learning the mean and covariance functions hyperparameters to the collected data. This can be performed either by maximizing the data's marginal likelihood or defining priors over the hyperparameters and sampling from the posterior using sampling (see Snoek et al. [2012] for more details).

Algorithm 1 SMBO Hyperparameter Optimization with GPs

```

 $\mathcal{R} \leftarrow \{\}$ 
for  $k \in \{1, 2, \dots, M\}$  do
   $\gamma \leftarrow \text{SMBO}(\mathcal{R})$  {Fit GP and maximize EI}
   $h_\gamma \leftarrow \mathcal{A}_\gamma(T)$  {Train with suggested  $\gamma$ }
   $\mathcal{R} \leftarrow \mathcal{R} \cup \{(\gamma, R_S(h_\gamma))\}$  {Add to collected data}
end for
 $\gamma^* \leftarrow \underset{(\gamma, R_S(h_\gamma)) \in \mathcal{R}}{\text{argmin}} R_S(h_\gamma)$ 
return  $h_{\gamma^*}$ 

```

While SMBO hyperparameter optimization can produce very good predictors, it can also suffer from overfitting on the validation set, especially for high-dimensional hyperparameter spaces. This is in part why an ensemble of predictors are often preferable in practice. Properly extending SMBO to the construction of ensembles is, however, not obvious. Here, we propose one such successful extension, building on the framework of Agnostic Bayes learning, described in the next section.

3 AGNOSTIC BAYES

In this section, we offer a brief overview of the Agnostic Bayes learning paradigm presented in Lacoste et al. [2014] and serving as a basis for the algorithm we present in this paper. Agnostic Bayes learning was used in Lacoste et al. [2014] as a framework for successfully constructing ensembles when the number of predictors in \mathcal{H} (i.e. the potential hyperparameter configurations Γ) was constrained to be finite (e.g. by restricting the space to a grid). In our context, we can thus enumerate the possible hyperparameter configurations from γ_1 to $\gamma_{|\Gamma|}$. This paper will generalize this approach to the infinite case later.

Agnostic Bayes learning attempts to directly address the problem of inferring what is the *best* function h^* in \mathcal{H} , according to the loss function \mathcal{L} . It infers a posterior $p_{h^*}(h_\gamma|S)$, i.e. a distribution over how likely each member of \mathcal{H} is the best predictor. This is in contrast with standard Bayesian learning, which implicitly assumes that \mathcal{H} contains the true data-generating model and infers a distribution for how likely each member of \mathcal{H} has generated the data (irrespective of what the loss \mathcal{L} is). From $p_{h^*}(h_\gamma|S)$, by marginalizing h^* , we obtain a probabilistic estimate for the best prediction $y^* \stackrel{\text{def}}{=} h^*(x)$

$$p_{y^*}(y|x, S) = \sum_{\gamma \in \Gamma} p_{h^*}(h_\gamma|S) I[h_\gamma(x) = y].$$

Finally, to commit to a final prediction, for a given x , we use the most probable answer². This yields the following ensemble decision rule

$$E^*(x) \stackrel{\text{def}}{=} \underset{y \in \mathcal{Y}}{\text{argmax}} p_{y^*}(y|x, S). \quad (4)$$

To estimate $p_{h^*}(h_\gamma|S)$, Agnostic Bayes learning uses the set of losses $l_{\gamma,i} \stackrel{\text{def}}{=} \mathcal{L}(h_\gamma(x_i), y_i)$ of each example $(x_i, y_i) \in S$ as evidence for inference. In Lacoste et al. [2014], a few different approaches are proposed and analyzed. A general strategy is to assume a joint prior $p(\mathbf{r})$ over the risks $r_\gamma \stackrel{\text{def}}{=} R_D(h_\gamma)$ of all possible hyperparameter configurations and choose a joint observation $p(l_{\gamma,i} \forall \gamma \in \Gamma | \mathbf{r})$ for the losses. From Bayes rule, we obtain the posterior $p(\mathbf{r}|S)$ from which we can compute

$$p_{h^*}(h_\gamma|S) = \mathbb{E}_{\mathbf{r}} [I[r_\gamma < r_{\gamma'}, \forall \gamma' \neq \gamma] | S] \quad (5)$$

with a Monte Carlo estimate. This would result in repeatedly sampling from $p(\mathbf{r}|S)$ and counting the number of times each γ has the smallest sampled risk r_γ to estimate $p_{h^*}(h_\gamma|S)$. Similarly, samples from $p_{h^*}(h_\gamma|S)$ could be obtained by sampling a risk vector \mathbf{r} from $p(\mathbf{r}|S)$ and returning the predictor h_γ with the lowest sampled risk. The

²As noted in Lacoste et al. [2014], $p_{y^*}(y|x, S)$ does not correspond to the probability of observing y given x and cannot be used with the optimal Bayes theory, thus justifying the usage of the most probable answer

ensemble decision rule of Equation 4 could then be implemented by repeatedly sampling from $p_{h^*}(h_\gamma|S)$ to construct the ensemble of predictors and using their average as the ensemble’s prediction.

Among the methods explored in Lacoste et al. [2014] to obtain samples from $p(\mathbf{r}|S)$, the bootstrap approach stands out for its efficiency and simplicity. Namely, to obtain a sample from $p(\mathbf{r}|S)$, we sample with replacement from S to obtain S' and return the vector of empirical risks $[R_{S'}(h_{\gamma_1}), \dots, R_{S'}(h_{\gamma_{|\Gamma|}})]^\top$ as a sample. While bootstrap only serves as a “poor man’s” posterior, it can be shown to be statistically related to a proper model with Dirichlet priors and its empirical performance was shown to be equivalent [Lacoste et al., 2014].

When the bootstrap method is used to obtain samples from $p_{h^*}(h_\gamma|S)$, the complete procedure for generating each ensemble member can be summarized by

$$\widetilde{h^*} = \underset{\gamma \in \Gamma}{\operatorname{argmin}} R_{S'}(h_\gamma), \quad (6)$$

where $\widetilde{h^*}$ corresponds to a sample from $p_{h^*}(h_\gamma|S)$. In this work, we use SMBO to address the optimization part. Thus, we can now extended to an uncountable set Γ .

This method can be seen as applying bagging on the validation set instead of the training set. However, we stand by the Agnostic Bayes theory since it offers a strong theoretical backbone to bagging as well as few refinements. Most importantly, the normal assumption of Section 3.3 in Lacoste et al. [2014] suggests that methods based on the covariance of the predictions such as ensemble pruning [Zhang et al., 2006] and MinCq [Roy et al., 2011] are simply different approximations of this idea. This connection allows us to be confident that the fast and simple algorithm we propose in this paper is at least equivalent in generalization performance to other state of the art ensemble methods. Finally, this claim is supported by the strong experimental section of Lacoste et al. [2014].

4 AGNOSTIC BAYES ENSEMBLE WITH SMBO

We now present our proposed method for automatically constructing an ensemble, without having to restrict Γ (or, equivalently \mathcal{H}) to a finite subset of hyperparameters.

As described in Section 3, to sample a predictor from the Agnostic Bayes bootstrap method, it suffices to obtain a bootstrap S' from S and solve the optimization problem of Equation 6. In our context where \mathcal{H} is possibly an infinite set of models trained on the training set T for any hyperparameter configuration γ , Equation 6 corresponds in fact to hyperparameter optimization where the holdout set is S' instead of S .

This suggests a simple procedure for building an ensemble of N predictors according to Agnostic Bayes i.e., that reflects our uncertainty about the true best model h^* . We could repeat the full SMBO hyperparameter optimization process N times, with different bootstrap S'_j , for $j \in \{1, 2, \dots, N\}$. However, for large ensembles, performing N runs of SMBO can be computationally expensive, since each run would need to train its own sequence of models.

We can notice however that predictors are always trained on the same training set T , no matter in which run of SMBO they were trained on. We propose a handy trick that exploits this observation to greatly accelerate the construction of the ensemble by almost a factor of N . Specifically, we propose to simultaneously optimize all N problems in a round-robin fashion. Thus, we maintain N different histories of evaluation \mathcal{R}_j , for $j \in \{1, 2, \dots, N\}$ and when a new predictor $h_\gamma = \mathcal{A}_\gamma(T)$ is obtained, we update all \mathcal{R}_j with $(\gamma, R_{S'_j}(h_\gamma))$. Notice that the different histories \mathcal{R}_j contain the empirical risks on different bootstrap holdout sets, but they are all updated at the cost of training only a single predictor. Also, to avoid recalculating multiple times $\mathcal{L}(h_\gamma(x_i), y_i)$, these values can be cached and shared in the computation of each \mathcal{R}_j . This leaves the task of updating all \mathcal{R}_j insignificant compared to the computational time usually required for training a predictor. This procedure is detailed in Algorithm 2.

Algorithm 2 Agnostic Bayes Ensemble with SMBO

```

for  $j \in \{1, 2, \dots, N\}$  do
   $\mathcal{R}_j \leftarrow \{\}$ 
   $S'_j \leftarrow \text{bootstrap}(S)$ 
end for

 $\mathcal{E} \leftarrow \{\}$  {Will contain all trained predictors}
for  $k \in \{1, 2, \dots, M\}$  do
   $v \leftarrow (k - 1) \text{ modulo } N + 1$ 
   $\gamma \leftarrow \text{SMBO}(\mathcal{R}_v)$ 
   $h_\gamma \leftarrow \mathcal{A}_\gamma(T)$ 
   $\mathcal{E} \leftarrow \mathcal{E} \cup \{h_\gamma\}$ 
  for  $j \in \{1, 2, \dots, N\}$  do
     $\mathcal{R}_j \leftarrow \mathcal{R}_j \cup \left\{ \left( \gamma, R_{S'_j}(h_\gamma) \right) \right\}$ 
  end for
end for

 $\mathcal{H}' \leftarrow \{\}$  {Will contain  $N$  selected predictors}
for  $j \in \{1, 2, \dots, N\}$  do
   $h_j \leftarrow \underset{h_\gamma \in \mathcal{E}}{\operatorname{argmin}} R_{S'_j}(h_\gamma)$ 
   $\mathcal{H}' \leftarrow \mathcal{H}' \cup \{h_j\}$ 
end for

 $p_{h^*}(h_\gamma|S) = \text{Uniform}(\mathcal{H}')$ 
return  $p_{h^*}(h_\gamma|S)$ 

```

By updating all \mathcal{R}_j at the same time, we *trick* each SMBO run by updating its history with points it did not suggest. This implies that the GP model behind each SMBO run will be able to condition on more observations than it would if the runs had been performed in isolation. This can only benefit the GPs and improve the quality of their suggestions.

While Algorithm 2 is sequential, it can be easily adapted to the parallelized version of SMBO presented in Snoek et al. [2012]. Also, it can be extended to use cross validation, based on the method developed in [Lacoste et al., 2014].

In our experiments, we fix $N = \lfloor \frac{M}{2} \rfloor$. This maximizes the number of samples used to estimate $p_{y^*}(y|x, S)$ while ensuring at least one SMBO step with a reasonably large history for each bootstrap. When the prediction time on the test set is a concern, we suggest to choose $N \approx 10$. We observed that it was usually enough to obtain most of the generalization gain.

Finally, since $p_{y^*}(y|x, S)$ is only estimated, finding the maximum, as requested in Equation 4, requires some form of density estimation. In the case of classification, we simply use the most probable class. However, in the regression case, we fit a normal distribution³. Thus, the maximum coincide with the average prediction. For a more complex \mathcal{Y} , such as in structured output tasks, we recommend to use an appropriate density estimation and increase the number of samples N .

5 RELATED WORK

In the Bayesian learning literature, a common way of dealing with hyperparameters in probabilistic predictors is to define hyperpriors and perform posterior inference to integrate them out. This process often results in also constructing an ensemble of predictors with different hyperparameters, sampled from the posterior. Powerful MCMC methods have been developed in order to accommodate for different types of hyperparameter spaces, including infinite spaces.

However, this approach requires that the family of predictors in question be probabilistic in order to apply Bayes rule. Moreover, even if the predictor family is probabilistic, the construction of the ensemble will entirely ignore the nature of the loss function that determines the measure of performance. The comparative advantage of the proposed Agnostic Bayes SMBO approach is thus that it can be used for any predictor family (probabilistic or not) and is loss-sensitive.

On the other hand, traditional ensemble methods such as Laviolette et al. [2011], Kim and Ghahramani [2012], and

³It is also possible to use a more elaborated method, such as kernel density estimation.

Zhang et al. [2006] require a predefined set of models and are not straightforward to adapt to an infinite set of models.

6 EXPERIMENTS

We now compare the SMBO ensemble approach (ESMBO) to three alternative methods for building a predictor from a machine learning algorithm with hyperparameters:

- A single model, whose hyperparameters were selected by hyperparameter optimization with SMBO (SMBO).
- A single model, whose hyperparameters were selected by a randomized search (RS), which in practice is often superior to grid search [Bergstra and Bengio, 2012].
- An Agnostic Bayes ensemble constructed from a randomly selected set of hyperparameters (ERS).

Both ESMBO and SMBO used GP models of the hold-out risk, with hyperparameters trained to maximize the marginal likelihood. A constant was used for the mean function, while the Matérn 5/2 kernel was used for the covariance function, with length scale parameters. The GP’s parameters were obtained by maximizing the marginal likelihood and a different length scale was used for each dimension⁴.

Each method is allowed to evaluate 150 hyperparameter configurations. To compare their performances, we perform statistical tests on several different hyperparameter spaces over two different collections of data sets.

6.1 HYPERPARAMETER SPACES

Here, we describe the hyperparameter spaces of all learning algorithms we employ in our experiments. Except for a custom implementation of the multilayer perceptron, we used scikit-learn⁵ for the implementation of all other learning algorithms.

Support Vector Machine We explore the soft margin parameter C for values ranging from 10^{-2} to 10^3 on a logarithmic scale. We use the RBF kernel $K(x, x') = e^{\gamma \|x - x'\|^2}$ and explore values of γ ranging from 10^{-5} to 10^3 on a logarithmic scale.

Support Vector Regressor We also use the RBF kernel and we explore the same values as for the Support Vector Machine. In addition, we explore the ϵ -tube parameter [Drucker et al., 1997] for values ranging between 10^{-2} and 1 on a logarithmic scale.

⁴We used the implementation provided by spearmint: <https://github.com/JasperSnoek/spearmint>

⁵<http://scikit-learn.org/>

Random Forest We fix the number of trees to 100 and we explore two different ways of producing them: either the original Breiman [2001] method or the extremely randomized trees method of Geurts et al. [2006]. We also explore the choice of bootstrapping or not the training set before generating a tree. Finally, the ratio of randomly considered features at each split for the construction of the trees is varied between 10^{-4} and 1 on a linear scale.

Gradient Boosted Classifier This is a tree-based algorithm using boosting [Friedman, 2001]. We fix the set of weak learners to 100 trees and take the maximum depth of each tree to be in $\{1, 2, \dots, 15\}$. The learning rate ranges between 10^{-2} and 1 on a logarithmic scale. Finally, the ratio of randomly considered features at each split for the construction of the trees varies between 10^{-3} and 1 on a linear scale.

Gradient Boosted Regressor We use the same parameters as for Gradient Boosted Classifier except that we explore a convex combination of the least square loss function and the least absolute deviation loss function. We also fix the ratio of considered features at each split to 1.

Multilayer Perceptron We use a 2 hidden layers perceptron with tanh activation function and a softmax function on the last layer. We minimize the negative log likelihood using the L-BFGS algorithm. Thus there is no learning rate parameter. However, we used a different L2 regularizer weight for each of the 3 layers with values ranging from 10^{-5} to 100 on a logarithmic scale. Also, the number of neurons on each layer can take values in $\{1, 2, \dots, 100\}$. In total, this yields a 5 dimensional hyperparameter space.

6.2 COMPARING METHODS ON MULTIPLE DATA SETS

To assess the generalization performances, we use a separate test set S^{test} , which is obtained by randomly partitioning the original data set. More precisely, we use the ratios 0.4, 0.3, and 0.3 for T , S and S^{test} respectively⁶. However, testing on a single data set is insufficient to testify the quality of a method that is meant to work across different tasks. Hence, we evaluate our methods on several data sets using metrics that do not assume commensurability across tasks [Demšar, 2006]. The metrics of choice are thus the expected rank and the pairwise winning frequency. Let $\mathcal{A}_i(T_j, S_j)$ be either one of our $K = 4$ model selection/ensemble construction algorithms run on the j^{th} data set, with training set T_j and validation set S_j . When comparing K algorithms, the rank of (best or ensemble)

predictor $h_i = \mathcal{A}_i(T_j, S_j)$ on test set S_j^{test} is defined as

$$\text{Rank}_{h_i, S^{\text{test}}} \stackrel{\text{def}}{=} \sum_{l=1}^K I \left[R_{S_j^{\text{test}}}(h_l) \leq R_{S_j^{\text{test}}}(h_i) \right].$$

Then, the expected rank of the i^{th} method is obtained from the empirical average over the L data sets i.e., $\mathbf{E}[\mathbf{R}]_i \stackrel{\text{def}}{=} \frac{1}{L} \sum_{j=1}^L \text{Rank}_{h_i, S_j^{\text{test}}}$. When comparing algorithm \mathcal{A}_i against algorithm \mathcal{A}_l , the winning frequency⁷ of \mathcal{A}_i is

$$\rho_{i,l} \stackrel{\text{def}}{=} \frac{1}{L} \sum_{j=1}^L I[R_{S_j^{\text{test}}}(h_i) < R_{S_j^{\text{test}}}(h_l)]$$

In the case of the expected rank, lower is better and for the winning frequency, it is the converse. Also, when $K = 2$, $\mathbf{E}[\mathbf{R}]_i = 1 + (1 - \rho_{i,l})$.

When the winning frequency $\rho_{i,l} > 0.5$, we say that method \mathcal{A}_i is better than method \mathcal{A}_l . However, to make sure that this is not the outcome of chance, we use statistical tests such as the sign test and the Poisson Binomial test (PB test) [Lacoste et al., 2012]. The PB test derives a posterior distribution over $\rho_{i,l}$ and integrates the probability mass above 0.5, denoted as $\Pr(A \succ B)$. When $\Pr(A \succ B) > 0.9$, we say that it is significant. Similarly for the sign test, when the p -value is lower than 0.05, it corresponds to a significant result. To report more information, we also use other thresholds for lightly significant and highly significant as described in Table 1.

To build a substantial collection of data sets, we used the AYSU collection [Ulař et al., 2009] coming from the UCI and the Delve repositories and we added the MNIST data set. We also converted the multiclass data sets to binary classification by either merging classes or selecting pairs of classes. The resulting benchmark contains 39 data sets. We have also collected 22 regression data sets from the Louis Torgo collection⁸.

6.3 TABLE NOTATION

The result tables present the winning frequency for each pair of methods, where grayed out values represent redundant information. As a complement, we also add the expected rank of each method in the rightmost column and sort the table according to this metric. To report the conclusion of the sign test and the PB test, we use different symbols to reflect different level of significance. The exact notation is presented in Table 1. The first symbol reports the result of the PB test and the second one, the sign test. For more stable results, we average the values obtained during the last 15 iterations.

⁷We deal with ties by attributing 0.5 to each method except for the sign test where the sample is simply discarded.

⁸These data sets were obtained from the following source : <http://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html>

⁶Is also possible to perform cross-validation as mentioned in Lacoste et al. [2014].

Table 1: Significance Notation Used in Result Tables.

Meaning	Symbol	$\Pr(A \succ B)$	p-value
Lightly significant	◦	> 0.8	< 0.1
Significant	•	> 0.9	< 0.05
Highly significant	•	> 0.95	< 0.01

Table 2: Pairwise Win Frequency For the 3 Different Regression Hyperparameter Spaces (Refer to Section 6.3 for the notation).

Support Vector Regressor					
	ESMBO	ERS	SMBO	RS	E[rank]
ESMBO	0.50	0.66 [•]	0.82 ^{••}	0.86 ^{••}	1.66
ERS	0.34 [◦]	0.50 [◦]	0.50 [◦]	0.77 ^{••}	2.38
SMBO	0.18 [◦]	0.50 [◦]	0.50 [◦]	0.64 [◦]	2.68
RS	0.14 [◦]	0.23 [◦]	0.36 [◦]	0.50 [◦]	3.27

Gradient Boosting Regressor					
	ERS	ESMBO	RS	SMBO	E[rank]
ERS	0.50	0.52	0.77 ^{••}	0.86 ^{••}	1.84
ESMBO	0.48 [◦]	0.50 [◦]	0.77 ^{••}	0.91 ^{••}	1.85
RS	0.23 [◦]	0.23 [◦]	0.50 [◦]	0.42 [◦]	3.12
SMBO	0.14 [◦]	0.09 [◦]	0.58 [◦]	0.50 [◦]	3.19

Random Forest					
	ESMBO	ERS	SMBO	RS	E[rank]
ESMBO	0.50	0.53	0.76 ^{••}	0.91 ^{••}	1.80
ERS	0.47 [◦]	0.50 [◦]	0.72 ^{••}	1.00 ^{••}	1.81
SMBO	0.24 [◦]	0.28 [◦]	0.50 [◦]	0.66 [◦]	2.82
RS	0.09 [◦]	0.00 [◦]	0.34 [◦]	0.50 [◦]	3.57

6.4 ANALYSIS

Looking at the overall results over 7 different hyperparameter spaces in Table 2 and Table 3, we observe that ESMBO is never significantly outperformed by any other method and often outperforms the others. More precisely, it is either ranked first or tightly following ERS. Looking more closely, we see that the cases where ESMBO does not significantly outperform ERS concerns hyperparameter spaces of low complexity. For example, most hyperparameter configurations of Random Forest yield good generalization performances. Thus, these cases do not require an elaborate hyperparameter search method. On the other hand, when looking at more challenging hyperparameter spaces such as Support Vector Regression and Multilayer Perceptrons, we clearly see the benefits of combining SMBO with Agnostic Bayes.

As described in Section 4, ESMBO is alternating between N different SMBO optimizations and deviates from the natural sequence of SMBO. To see if this aspect of ESMBO can influence its convergence rate, we present a temporal analysis of the methods in Figure 1 and Figure 2. The left columns depict $\Pr(A \succ B)$ for selected pairs of methods and the right columns present the expected rank of each method over time.

Table 3: Pairwise Win Frequency for the 4 Different Classification Hyperparameter Spaces (Refer to Section 6.3 for the notation).

Support Vector Machine					
	ESMBO	RS	SMBO	ERS	E[rank]
ESMBO	0.50	0.54	0.55	0.56	2.35
RS	0.46 [◦]	0.50 [◦]	0.51 [◦]	0.51 [◦]	2.52
SMBO	0.45 [◦]	0.49 [◦]	0.50 [◦]	0.53 [◦]	2.54
ERS	0.44 [◦]	0.49 [◦]	0.47 [◦]	0.50 [◦]	2.59

Gradient Boosting Classifier					
	ESMBO	ERS	RS	SMBO	E[rank]
ESMBO	0.50	0.51	0.59	0.65 ^{◦•}	2.25
ERS	0.49 [◦]	0.50 [◦]	0.59 [◦]	0.64 ^{◦◦}	2.28
RS	0.41 [◦]	0.41 [◦]	0.50 [◦]	0.55 [◦]	2.64
SMBO	0.35 [◦]	0.36 [◦]	0.45 [◦]	0.50 [◦]	2.83

Random Forest					
	ERS	ESMBO	RS	SMBO	E[rank]
ERS	0.50	0.52	0.60 [◦]	0.64 ^{◦•}	2.24
ESMBO	0.48 [◦]	0.50 [◦]	0.60 [◦]	0.67 ^{◦•}	2.25
RS	0.40 [◦]	0.40 [◦]	0.50 [◦]	0.57 [◦]	2.63
SMBO	0.36 [◦]	0.33 [◦]	0.43 [◦]	0.50 [◦]	2.89

Multilayer Perceptron					
	ESMBO	SMBO	ERS	RS	E[rank]
ESMBO	0.50	0.57 [◦]	0.76 ^{••}	0.75 ^{••}	1.92
SMBO	0.43 [◦]	0.50 [◦]	0.68 ^{◦•}	0.68 ^{◦•}	2.21
ERS	0.24 [◦]	0.32 [◦]	0.50 [◦]	0.54 [◦]	2.91
RS	0.25 [◦]	0.32 [◦]	0.46 [◦]	0.50 [◦]	2.96

A general analysis clearly shows that there is no significant degradation in terms of convergence speed. In fact, we generally observe the opposite. More precisely, looking at $\Pr(\text{ESMBO} \succ \text{SMBO})$, the green curve of the left columns, it usually reaches a significantly better state right at the beginning or within the first few iterations. A notable exception to that trend occurs with the Multilayer Perceptrons, where SMBO is significantly better than ESMBO for a few iterations at the beginning. Then, it gets quickly outperformed by ESMBO.

7 CONCLUSION

We described a successful method for automatically constructing ensembles without requiring hand-selection of models or a grid search. The method can adapt the SMBO hyperparameter optimization algorithm so that it can produce an ensemble instead of a single model. Theoretically, the method is motivated by an Agnostic Bayesian paradigm which attempts to construct ensembles that reflect the uncertainty over which a model actually has the smallest true risk. The resulting method is easy to implement and comes with no extra computational cost at learning time. Its generalization performance and convergence speed are also dominant according to experiments on 22 regression and 39 classification data sets.

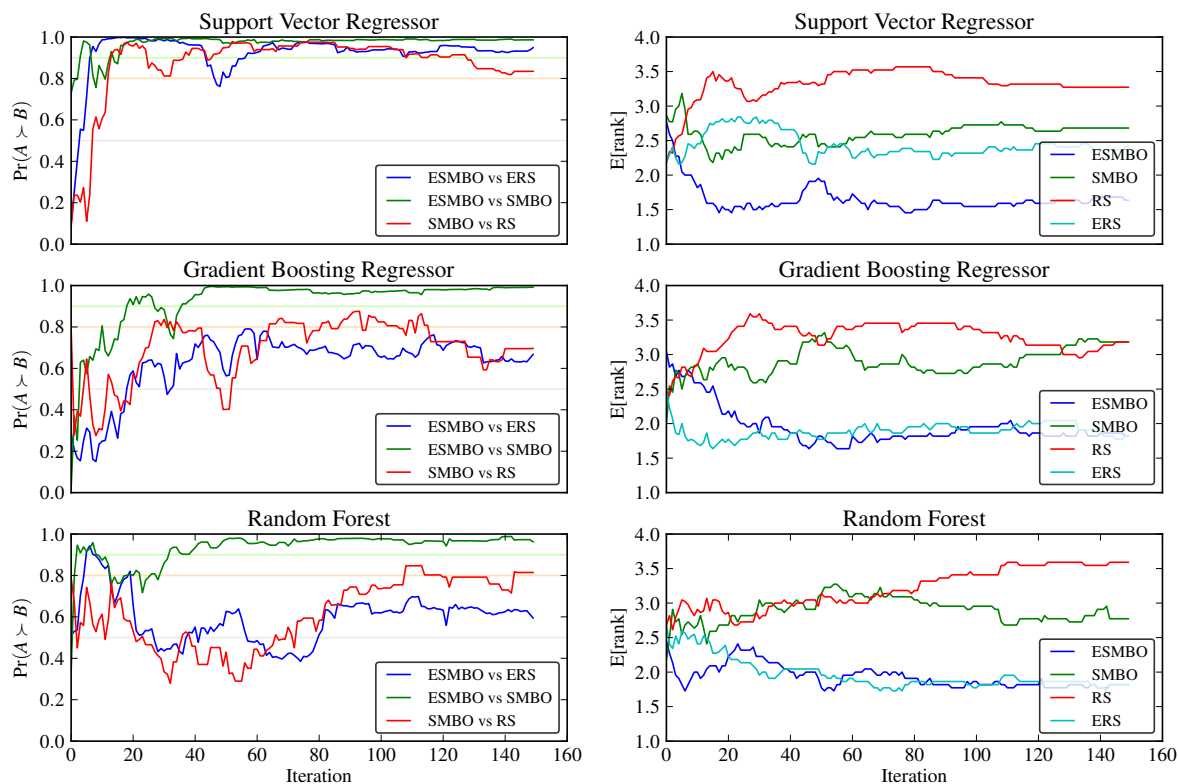


Figure 1: PB Probability and Expected Rank over Time for the 3 Regression Hyperparameter Spaces.

Acknowledgement

Thanks to Calcul Québec for providing support and access to Colosse’s high performance computer grid. This work was supported by NSERC Discovery Grants 122405 (M. M.), 262067 (F. L.) and 418327 (H. L.).

References

- James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *NIPS*, pages 2546–2554, 2011.
- Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Learning and Intelligent Optimization*, pages 507–523. Springer, 2011.
- Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical bayesian optimization of machine learning algorithms. In *NIPS*, pages 2960–2968, 2012.
- Chris Thornton, Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms. In *Proc. of KDD-2013*, pages 847–855, 2013.
- Robert M Bell, Yehuda Koren, and Chris Volinsky. The bellkor solution to the netflix prize. *KorBell Team’s Report to Netflix*, 2007.
- Alexandre Lacoste, Mario Marchand, François Laviolette, and Hugo Larochelle. Agnostic bayesian learning of ensembles. In *Proceedings of The 31st International Conference on Machine Learning*, pages 611–619, 2014.
- James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305, 2012.
- Yi Zhang, Samuel Burer, and W Nick Street. Ensemble pruning via semi-definite programming. *The Journal of Machine Learning Research*, 7:1315–1338, 2006.
- Jean-François Roy, François Laviolette, and Mario Marchand. From pac-bayes bounds to quadratic programs for majority votes. In *ICML*, pages 649–656, 2011.
- F. Laviolette, M. Marchand, and J.F. Roy. From pac-bayes bounds to quadratic programs for majority votes. *moment*, 1500:Q2, 2011.
- Hyun-Chul Kim and Zoubin Ghahramani. Bayesian classifier combination. *Journal of Machine Learning Research - Proceedings Track*, 22:619–627, 2012.
- Harris Drucker, Chris JC Burges, Linda Kaufman, Alex Smola, and Vladimir Vapnik. Support vector regression machines. *Advances in neural information processing systems*, pages 155–161, 1997.

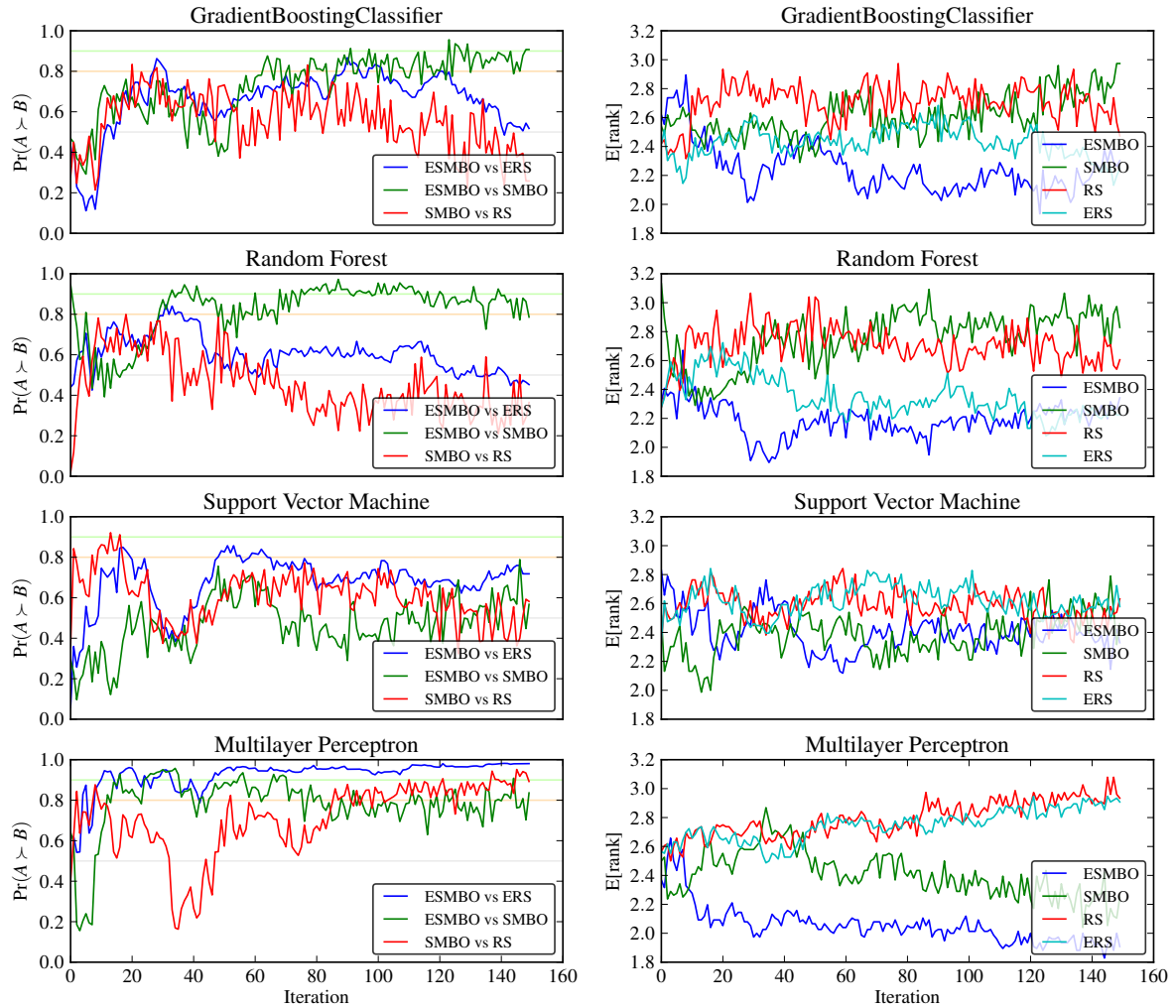


Figure 2: PB Probability and Expected Rank over Time for the 4 Classification Hyperparameter Spaces.

- L. Breiman. Random forests. *Machine learning*, 45(1): 5–32, 2001.
- Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine learning*, 63(1):3–42, 2006.
- Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, pages 1189–1232, 2001.
- Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.

Alexandre Lacoste, François Laviolette, and Mario Marchand. Bayesian comparison of machine learning algorithms on single and multiple datasets. *Journal of Machine Learning Research - Proceedings Track*, 22:665–675, 2012.

Aydın Ulaş, Murat Semerci, Olcay Taner Yıldız, and Ethem Alpaydın. Incremental construction of classifier and discriminant ensembles. *Information Sciences*, 179(9): 1298–1318, April 2009.

Position-Aware ListMLE: A Sequential Learning Process for Ranking

Yanyan Lan¹ Yadong Zhu² Jiafeng Guo¹ Shuzi Niu² Xueqi Cheng¹

Institute of Computing Technology, Chinese Academy of Sciences, Beijing, P. R. China

¹{lanyanyan, guojiafeng, cxq}@ict.ac.cn, ²{zhuyadong, niushuzi}@software.ict.ac.cn

Abstract

ListMLE is a state-of-the-art listwise learning-to-rank algorithm, which has been shown to work very well in application. It defines the probability distribution based on Plackett-Luce Model in a top-down style to take into account the position information. However, both empirical contradiction and theoretical results indicate that ListMLE cannot well capture the position importance, which is a key factor in ranking. To amend the problem, this paper proposes a new listwise ranking method, called position-aware ListMLE (p-ListMLE for short). It views the ranking problem as a sequential learning process, with each step learning a subset of parameters which maximize the corresponding stepwise probability distribution. To solve this *sequential multi-objective optimization* problem, we propose to use *linear scalarization* strategy to transform it into a single-objective optimization problem, which is efficient for computation. Our theoretical study shows that p-ListMLE is better than ListMLE in statistical consistency with respect to typical ranking evaluation measure NDCG. Furthermore, our experiments on benchmark datasets demonstrate that the proposed method can significantly improve the performance of ListMLE and outperform state-of-the-art listwise learning-to-rank algorithms as well.

1 INTRODUCTION

Ranking is an important problem in various applications, such as information retrieval, meta search and collaborative filtering. In recent years, machine learning technologies have been widely applied for ranking, and a new research branch named learning to rank has emerged. A learning-to-rank process can be described as follows. In training,

a number of sets (queries) of objects (documents) are given and within each set the objects are labeled by assessors, mainly based on multi-level ratings. The target of learning is to create a model that provides a ranking over the objects that best respects the observed labels. In testing, given a new set of objects, the trained model is applied to generate a ranking list of the objects. To evaluate the performance of a ranking system, many position-aware evaluation measures such as NDCG [9], MAP [2], ERR [5] are used to reflect users' bias on different positions. That is, users often care more about the results on top positions in a ranking [3, 16, 21].

In literature, pointwise algorithms such as McRank [14] were first proposed to solve the ranking problem, which transformed ranking into (ordinal) regression or classification on individual documents. The idea is natural but it is comprehensively criticized for using different objectives from ranking. Therefore, pairwise algorithms such as RankSVM [10], RankBoost [7] and RankNet [1] were then proposed to view a pair of items as the object, and transform ranking into the pairwise classification problem. However, the pairwise approach highly ignores the position information over different pairs, which is quite important for ranking as mentioned above. To overcome the weakness of pairwise ranking algorithms, listwise ranking algorithms such as ListMLE [21], ListNet [4], RankCosine [17] and AdaRank [22] were proposed, which view the whole ranking list as the object. For example, ListMLE utilized the likelihood loss of the probability distribution based on Plackett-Luce model for optimization. According to previous studies [4, 15, 17, 21], the listwise approach can outperform the other two approaches on benchmark datasets.

Seemingly listwise approaches can well solve the ranking problem by directly modeling the ranking lists and thus taking into account the position information. However, both empirical contradiction and theoretical results indicate that listwise approaches cannot well capture the position importance, which is a key factor in ranking [13, 6]. In this paper, we take the typical listwise method ListMLE as an example to illustrate this problem. Empirically, given two

ranking functions f_1 and f_2 , the error of f_1 occurs in the top positions and that of f_2 occurs in the bottom positions. We find that ListMLE will prefer f_1 to f_2 , leading to lower performance under the IR evaluation measure such as NDCG. Theoretically, it has been proven in [21] that ListMLE is consistent¹ with permutation level 0-1 loss, a loss without considering the importance of different positions.

We analyze the underlying reason behind these above results. We find that the probability in ListMLE is defined in a top-down style which seems to reflect the position importance in ranking. However, due to the chain rule of probability, the decomposition of probability in ListMLE is not unique, indicating that different positions are actually equally important in such definition. To amend this problem, we propose a new listwise ranking approach, namely position-aware ListMLE (p-ListMLE for short), which views ranking as a sequential learning process. Specifically, at step 1, it aims to maximize the top 1 probability distribution of Plackett-Luce model. At step i , it aims to maximize the i -th conditional probability distribution of Plackett-Luce model given the top $i-1$ items. To solve the sequential learning problem, we propose to transform it into a single-objective optimization problem, which is equivalent to minimizing a new surrogate loss.

Theoretically, we study the statistical consistency issue of p-ListMLE. Following the technique used in [11], we can prove that with RDPS as the assumption, p-ListMLE will be consistent with Weighted Pairwise Disagreement Loss (WPDL for short), which is equivalent to a certain NDCG. Further considering the previous result that ListMLE is consistent with permutation level 0-1 loss, we can see that p-ListMLE is better than ListMLE theoretically. We further conduct extensive experiments on benchmark datasets LETOR4.0, and the empirical results demonstrate that the proposed p-ListMLE can significantly outperform the original ListMLE as well as other state-of-the-art listwise learning-to-rank algorithms.

The contribution of this paper lies in the following aspects:

- (1) We provide a novel view of ranking as a sequential learning process, with each step to learn a subset of parameters which maximize the corresponding stepwise probability distribution;
- (2) We propose a new ranking algorithm to incorporate positions into the learning process;
- (3) We provide theoretical analysis on the consistency of the proposed ranking algorithm.

The remainder of the paper are organized as follows. In section 2, we provide some backgrounds on ListMLE, in-

cluding the framework of listwise learning to rank, the algorithm of ListMLE, and the theoretical results on ListMLE. Section 3 describes the motivation of this paper and section 4 presents our main results, including the novel sequential view of ranking and the loss function of the new algorithm. Section 5 and 6 presents our theoretical and experimental results, respectively. Section 7 concludes the paper.

2 BACKGROUNDS

In this section, we give some backgrounds on ListMLE, which is a famous listwise ranking algorithm. Listwise learning to rank addresses the ranking problem in the following way. In learning, it takes ranking lists of objects as instances and trains a ranking function through the minimization of a listwise loss function defined on predicted list and the ground-truth list. Following [21], we give the mathematical description of listwise learning-to-rank framework as follows.

2.1 Listwise Learning to Rank

Let $\mathbf{x} = \{x_1, \dots, x_n\} \in X$ be a set of objects to be ranked, and $\mathbf{y} = \{y_1, \dots, y_n\} \in Y$ be the ground-truth permutation of these objects, where y_i stands for the position of x_i and $\mathbf{y}^{-1}(i)$ stands for the index of items in the i -th position of \mathbf{y} . We assume that (\mathbf{x}, \mathbf{y}) are sampled according to a fixed but unknown joint probability distribution P_{XY} . Let $f : X \rightarrow \mathbb{R}^n$ be a ranking function, where \mathbb{R}^n denotes a n -dimensional real-valued vector space. The task of listwise learning to rank is to learn a ranking function that can minimize the *expected risk* $R_0(h)$, defined as:

$$R_0(h) = \int_{X \times Y} L_0(f; \mathbf{x}, \mathbf{y}) dP_{XY}(\mathbf{x}, \mathbf{y}),$$

where L_0 is a true loss of listwise learning to rank. For example, Xia et al. [21] utilized permutation level 0-1 loss as the true loss, which takes the following form.

$$L_{0-1}(f; \mathbf{x}, \mathbf{y}) = I_{\{\pi_f(\mathbf{x}) \neq \mathbf{y}\}}, \quad (1)$$

where $I_{\{\cdot\}}$ is an indicator function, with $I_A = 1$, if A is true, and $I_A = 0$, otherwise. π_f stands for the output permutation induced by sorting the objects in descending order of scores produced by f , that is,

$$\pi_f(\mathbf{x}) = \text{sort}(f(x_1), \dots, f(x_n)).$$

Since P_{XY} is unknown, the optimal ranking function which minimizes the expected loss cannot be easily obtained. In practice, we are usually given independently and identically distributed samples $S = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N \sim P_{XY}$, where $\mathbf{x}_i = (x_1^{(i)}, \dots, x_{n_i}^{(i)})$ and $\mathbf{y}_i = (y_1^{(i)}, \dots, y_{n_i}^{(i)})$.

¹Please note that Xia et al. [19] prove that a modification to ListMLE is consistent with top-k 0-1 loss, however, top-k 0-1 loss take the top k positions as equal, therefore cannot well capture the position importance.

Therefore, we instead try to obtain a ranking function that minimize the *empirical risk*.

$$\hat{R}(h) = \frac{1}{N} \sum_{i=1}^N L_0(h; \mathbf{x}_i, \mathbf{y}_i).$$

However, the true loss is usually nonconvex, which poses a challenge to the optimization of the empirical risk. As is done in the literature of machine learning, people usually use surrogate losses as an approximation of the true loss, and turn to minimize the corresponding *surrogate empirical risk* instead.

$$R_\phi(f, \mathbf{x}, \mathbf{y}) = \frac{1}{N} \sum_{i=1}^N L_\phi(f; \mathbf{x}_i, \mathbf{y}_i).$$

2.2 ListMLE

ListMLE is such a listwise ranking algorithms which utilize a likelihood loss as the surrogate loss, defined as follows.

$$L(f; \mathbf{x}, \mathbf{y}) = -\log P(\mathbf{y}|\mathbf{x}; f), \quad (2)$$

where,

$$P(\mathbf{y}|\mathbf{x}; f) = \prod_{i=1}^n \frac{\exp(f(x_{\mathbf{y}^{-1}(i)}))}{\sum_{k=i}^n \exp(f(x_{\mathbf{y}^{-1}(k)}))}. \quad (3)$$

The above probability is defined according to Plackett-Luce model. That is to say, the probability of a permutation is first decomposed to the product of a stepwise conditional probability, with the i -th conditional probability standing for the probability that the document is ranked at the i -th position given the top $i - 1$ objects are ranked correctly. The precise form are given in the following equations.

$$\begin{aligned} & P(\mathbf{y}|\mathbf{x}; f) \\ &= P(\mathbf{y}^{-1}(1), \mathbf{y}^{-1}(2), \dots, \mathbf{y}^{-1}(n)|\mathbf{x}; f) \\ &= P(\mathbf{y}^{-1}(1)|\mathbf{x}; f) \prod_{i=2}^n P(\mathbf{y}^{-1}(i)|\mathbf{x}, \mathbf{y}^{-1}(1), \dots, \mathbf{y}^{-1}(i-1); f) \end{aligned} \quad (4)$$

where,

$$P(\mathbf{y}^{-1}(1)|\mathbf{x}; f) = \frac{\exp(f(x_{\mathbf{y}^{-1}(1)}))}{\sum_{k=1}^n \exp(f(x_{\mathbf{y}^{-1}(k)}))}, \quad (5)$$

$$\begin{aligned} & P(\mathbf{y}^{-1}(i)|\mathbf{x}, \mathbf{y}^{-1}(1), \mathbf{y}^{-1}(2), \dots, \mathbf{y}^{-1}(i-1); f) \\ &= \frac{\exp(f(x_{\mathbf{y}^{-1}(i)}))}{\sum_{k=j}^n \exp(f(x_{\mathbf{y}^{-1}(k)}))}, \forall i = 2, \dots, n \end{aligned} \quad (6)$$

2.3 Consistency

Previous theoretical analyses on ListMLE were mainly focused on generalization and consistency. For generalization analysis, Lan et al. [12] has derived the generalization bounds of listwise ranking methods, including ListMLE [21], ListNet [4] and RankCosine [17]. As to the statistical consistency analysis, the loss function in ListMLE has been proven to be consistent with permutation level 0-1 loss [20], where consistency is defined as follows.

Definition 1. We say a surrogate loss L_ϕ is statistical consistent with respect to the true loss L_0 , if $\forall \epsilon_1 > 0, \exists \epsilon_2 > 0$, such that for any ranking function $f \in \mathcal{F}$, $R_\phi(f) \leq \inf_{h \in \mathcal{F}} R_\phi(h) + \epsilon_2$ implies $R_0(f) \leq \inf_{h \in \mathcal{F}} R_0(h) + \epsilon_1$.

Statistical consistency is a desired property for a good surrogate loss, which measures whether the expected true risk of the ranking function obtained by minimizing a surrogate loss converges to the expected true risk of the optimal ranking in the large sample limit. Therefore, the consistency of ListMLE with respect to permutation level 0-1 loss means that the the surrogate loss of ListMLE is a good surrogate of permutation level 0-1 loss theoretically. However, 0-1 loss is not a ‘good’ loss for the ranking problem, since it largely ignores the impact of positions, which is crucial in ranking. Therefore, in this paper, we propose to study how to improve ListMLE to make it consistent with a better loss for ranking.

3 MOTIVATIONS

The motivation of this work comes from both empirical contradiction and theoretical results in ListMLE, indicating that position importance, which is a key factor in ranking, is actually ignored in this listwise approach.

3.1 Empirical Contradiction

Firstly, we take a case study on some toy data to show that position importance is actually not considered in ListMLE.

We are given a set of documents $\mathbf{x} = \{x_1, \dots, x_5\}$ and their ground-truth labels $\mathbf{z} = (z_1, \dots, z_5)$ in terms of 5-level ratings, where $z_i = i$. That is to say, the best ranking list is $\mathbf{y} = (1, 2, 3, 4, 5)$. Suppose we are given two ranking functions f_1 and f_2 with corresponding scores $\mathbf{y}_1 = (\ln 4, \ln 5, \ln 3, \ln 2, \ln 1)$ and $\mathbf{y}_2 = (\ln 5, \ln 4, \ln 1, \ln 2, \ln 3)$, where $\mathbf{y}_i(j) = f_i(x_j)$. As we can see, the first two documents are mistakenly ranked by f_1 , while the last three documents are mistakenly ranked by f_2 . The corresponding likelihood losses of these two rank-

ing functions in ListMLE are listed as follows.

$$\begin{aligned}
L(f_1, \mathbf{x}, \mathbf{y}) &= -\log \left(\frac{4}{4+5+3+2+1} \cdot \frac{5}{5+3+2+1} \cdot \frac{3}{3+2+1} \cdot \frac{2}{2+1} \cdot \frac{1}{1} \right), \\
L(f_2, \mathbf{x}, \mathbf{y}) &= -\log \left(\frac{5}{5+4+3+2+1} \cdot \frac{4}{4+3+2+1} \cdot \frac{1}{1+2+3} \cdot \frac{2}{2+3} \cdot \frac{3}{3} \right).
\end{aligned}$$

Through comparison, we can see that f_1 is better than f_2 in terms of likelihood loss of ListMLE, as indicated by the fact that $L(f_1, \mathbf{x}, \mathbf{y}) < L(f_2, \mathbf{x}, \mathbf{y})$.

However, from the view of IR evaluation measure NDCG, we can see that $NDCG(f_1, \mathbf{x}, \mathbf{y}) < NDCG(f_2, \mathbf{x}, \mathbf{y})$, showing that f_2 should be preferred to f_1 .

$$\begin{aligned}
NDCG@5(f_1, \mathbf{x}, \mathbf{y}) &= \frac{1}{N_5} \left((2^5 - 1) \log \frac{1}{3} + (2^4 - 1) \log \frac{1}{2} + (2^3 - 1) \log \frac{1}{4} \right) \\
&\quad + \frac{1}{N_5} \left((2^2 - 1) \log \frac{1}{5} + (2^1 - 1) \log \frac{1}{6} \right), \\
NDCG@5(f_2, \mathbf{x}, \mathbf{y}) &= \frac{1}{N_5} \left((2^5 - 1) \log \frac{1}{2} + (2^4 - 1) \log \frac{1}{3} + (2^3 - 1) \log \frac{1}{6} \right) \\
&\quad + \frac{1}{N_5} \left((2^2 - 1) \log \frac{1}{5} + (2^1 - 1) \log \frac{1}{4} \right),
\end{aligned}$$

As we know, IR evaluation measures such as NDCG reflect the fact that users are more concerned on results in top positions in a ranking. Therefore, the mistakes in top positions will be more severe than that in low positions. The empirical contradiction between ListMLE and IR evaluation measures thus indicates that the loss of ListMLE cannot well capture the position importance.

3.2 Theoretical Result

In [21], it was proven that the loss functions of ListMLE [21] are consistent with permutation level 0-1 loss. However, permutation level 0-1 loss actually does not take position importance into account. Therefore, theoretical consistency between ListMLE and permutation level 0-1 loss also demonstrates that the position importance cannot be well captured by the loss of ListMLE.

4 POSITION-AWARE LISTMLE

The above results indicates that ListMLE ignores the position importance, which is a key factor for ranking. However, the probability in ListMLE is defined in a top-down style

which seems to reflect the position importance in ranking. This contradiction makes us revisit the algorithm of ListMLE. As we can see, the probability on permutation in ListMLE can be decomposed as follows.

$$\begin{aligned}
P(\mathbf{y}|\mathbf{x}; f) &= P(\mathbf{y}^{-1}(1), \mathbf{y}^{-1}(2), \dots, \mathbf{y}^{-1}(n)|\mathbf{x}; f) \\
&= P(\mathbf{y}^{-1}(1)|\mathbf{x}; f) \prod_{i=2}^n P(\mathbf{y}^{-1}(i)|\mathbf{x}, \mathbf{y}^{-1}(1), \dots, \mathbf{y}^{-1}(i-1); f)
\end{aligned} \tag{7}$$

However, the *chain rule of probability* described as follows tells us that this is not the unique decomposition.

$$\begin{aligned}
P(A_1, \dots, A_n) &= P(A_{i_1})P(A_{i_2}|A_{i_1}) \dots P(A_{i_n}|A_{i_1}, \dots, A_{i_{n-1}}),
\end{aligned}$$

where (i_1, \dots, i_n) is any permutation of $(1, \dots, n)$.

As a consequence, the probability is equal to any decomposition described as follows.

$$\begin{aligned}
P(\mathbf{y}|\mathbf{x}; f) &= P(\mathbf{y}^{-1}(1), \mathbf{y}^{-1}(2), \dots, \mathbf{y}^{-1}(n)|\mathbf{x}; f) \\
&= P(\mathbf{y}^{-1}(i_1)|\mathbf{x}; f) \prod_{j=2}^n P(\mathbf{y}^{-1}(i_j)|\mathbf{x}, \mathbf{y}^{-1}(i_1), \dots, \mathbf{y}^{-1}(i_{j-1}); f)
\end{aligned} \tag{8}$$

where (i_1, \dots, i_n) is any permutation of $(1, \dots, n)$. In this way, different positions are actually equally important under the probability definition of ListMLE. In other words, the loss of ListMLE cannot reflect the position importance in a top-down style. Therefore, we propose a new listwise ranking approach, namely p-ListMLE, to capture the position importance in a ‘true’ top-down style.

4.1 Ranking As a Sequential Process

To reflect the position importance in a top-down style, i.e. higher position is more important, we propose a new sequential learning process for ranking as follows.

Step 1: Maximizing the probability that the top 1 object is selected with mathematical description as follows.

$$\max_{f \in \mathcal{F}} P(\mathbf{y}^{-1}(1)|\mathbf{x}; f);$$

Step i: For $i = 2, \dots, n$, we denote the subset of ranking functions that reach the maximum in Step $i - 1$ as S_{i-1} . The task of step i is to maximize the probability that the object with position i in ground-truth permutation is selected given the top $i - 1$ objects ranked correctly. The mathematical formulation is described as follows.

$$\max_{f \in S_{i-1}} P(\mathbf{y}^{-1}(i)|\mathbf{x}, \mathbf{y}^{-1}(1), \dots, \mathbf{y}^{-1}(i-1); f);$$

Step n+1: The learning process ends, and the ranking function f is randomly selected from S_{n-1} as the output ranking function.

4.2 Loss Function

In order to solve the above *sequential multi-objective optimization* problem, we propose to use *linear scalarization* strategy [8] to transform it into a single-objective optimization problem, and emphasize the early steps to reflect the position importance.

$$\begin{aligned} \min_{f \in \mathcal{F}} \Phi(f), \\ \Phi(f) = -\sum_{i=2}^n \alpha(i) \log P(\mathbf{y}^{-1}(i) | \mathbf{x}, \mathbf{y}^{-1}(1), \dots, \mathbf{y}^{-1}(i-1); f) \\ - \alpha(1) \log P(\mathbf{y}^{-1}(1) | \mathbf{x}; f), \end{aligned}$$

where $\alpha(\cdot)$ is a decreasing function, i.e. $\alpha(i) > \alpha(i+1)$.

Incorporating the probability based on Plackett-Luce model as described in Eq. (5) and Eq. (6) into the above optimization problem, we obtain a new algorithm which minimizes the following likelihood loss function for p-ListMLE.

$$\begin{aligned} L_p(f; \mathbf{x}, \mathbf{y}) \\ = \sum_{i=1}^n \alpha(i) (-f(x_{\mathbf{y}^{-1}(i)}) + \log(\sum_{j=i}^n \exp(f(x_{\mathbf{y}^{-1}(j)})))) \end{aligned} \quad (9)$$

4.3 Case Revisit

Here, we revisit the case used in section 3.1 to show that after introducing position factor α into ListMLE, the empirical contradiction will disappear. We first compute the likelihood losses of p-ListMLE with respect to f_1 and f_2 , which are listed as follows.

$$\begin{aligned} L_p(f_1, \mathbf{x}, \mathbf{y}) \\ = -\alpha(1) \log\left(\frac{4}{4+5+3+2+1}\right) - \alpha(2) \log\left(\frac{5}{5+3+2+1}\right) \\ - \alpha(3) \log\left(\frac{3}{3+2+1}\right) - \alpha(4) \log\left(\frac{2}{2+1}\right), \end{aligned}$$

$$\begin{aligned} L_p(f_2, \mathbf{x}, \mathbf{y}) \\ = -\alpha(1) \log\left(\frac{5}{5+4+3+2+1}\right) - \alpha(2) \log\left(\frac{4}{4+3+2+1}\right) \\ - \alpha(3) \log\left(\frac{1}{1+2+3}\right) - \alpha(4) \log\left(\frac{2}{2+3}\right). \end{aligned}$$

In this way, the following equality holds:

$$\begin{aligned} L_p(f_1, \mathbf{x}, \mathbf{y}) - L_p(f_2, \mathbf{x}, \mathbf{y}) \\ = (\alpha(1) - \alpha(2) - \alpha(4)) \ln 5 - (\alpha(1) - \alpha(2)) \ln 4 \\ - (\alpha(3) - \alpha(4)) \ln 3 + \alpha(2)(\ln 11 - \ln 10) \\ > (\alpha(1) - \alpha(2) - \alpha(4)) \ln 5 - (\alpha(1) - \alpha(2)) \\ + \alpha(3) - \alpha(4)) \ln 4, \end{aligned}$$

Therefore, as long as α satisfies the following condition in Eq. (10), we will have $L_p(f_1, \mathbf{x}, \mathbf{y}) > L_p(f_2, \mathbf{x}, \mathbf{y})$.

$$\frac{\alpha(1) - \alpha(2) - \alpha(4)}{\alpha(1) - \alpha(2) + \alpha(3) - \alpha(4)} > \frac{\ln 4}{\ln 5}. \quad (10)$$

That is, f_2 is preferred to f_1 in terms of p-ListMLE, which is consistent with *NDCG* as shown in Section 3.1. This condition is easy to be satisfied as long as $\alpha(1)$ is far larger than the other $\alpha(i), i = 2, \dots, 5$.

5 STATISTICAL CONSISTENCY OF P-LISTMLE

In this section, we study the statistical consistency of p-ListMLE. In [21], it is proved that ListMLE is consistent with permutation level 0-1 loss. Since position factor $\alpha(\cdot)$ is introduced in p-ListMLE, we consider weighted pairwise loss (WPD L) defined in [11] as the true loss.

$$L_{\text{WPD L}}(f; \mathbf{x}, \mathbf{y}) = \sum_{i,j: r_i > r_j} D(r_i, r_j) I_{\{f(x_i) - f(x_j) \leq 0\}}, \quad (11)$$

where $r_i = n - y_i$, $D(r_i, r_j) = \alpha(r_j) - \alpha(r_i) > 0$.

Firstly, we introduce the definition of a rank-differentiable probability space (RDPS for short), with which we can prove that p-ListMLE is consistent with WPD L. Hereafter, we will also refer to data from RDPS as having a rank-differentiable property.

5.1 A Rank-Differentiable Probability Space

Before introducing the definition of RDPS, we give two definitions, *an equivalence class of ratings* and *dual ratings*. Intuitively, we say two ratings are equivalent if they induce the same ranking or preference relationships. Meanwhile, we say two ratings are the dual ratings with respect to a pair of objects, if the two ratings only exchange the ratings of the two objects while keeping the ratings of other objects unchanged. The formal definitions are given as follows.

Definition 2. A ratings \mathbf{r} is called equivalent to $\tilde{\mathbf{r}}$, denoted as $\mathbf{r} \sim \tilde{\mathbf{r}}$, if $\mathcal{P}(\mathbf{r}) = \mathcal{P}(\tilde{\mathbf{r}})$. Where $\mathcal{P}(\mathbf{r}) = \{(i, j) : r_i > r_j\}$ and $\mathcal{P}(\tilde{\mathbf{r}}) = \{(i, j) : \tilde{r}_i > \tilde{r}_j\}$ stand for the preference relationships induced by \mathbf{r} and $\tilde{\mathbf{r}}$, respectively.

Therefore, an equivalence class of the ratings \mathbf{r} , denoted as $[\mathbf{r}]$, is defined as the set of ratings which are equivalent to \mathbf{r} . That is, $[\mathbf{r}] = \{\tilde{\mathbf{r}} \in \mathcal{R} : \tilde{\mathbf{r}} \sim \mathbf{r}\}$.

Definition 3. Let $R(i, j) = \{\mathbf{r} \in \mathcal{R} : r_i > r_j\}$, \mathbf{r}' is called the dual ratings of $\mathbf{r} \in R(i, j)$ with respect to (i, j) if $r'_j = r_i, r'_i = r_j, r'_k = r_k, \forall k \neq i, j$.

Now we give the definition of RDPS. An intuitive explanation on this definition is that there exists a unique equivalence class of ratings that for each induced pairwise preference relationship, the probability will be able to separate the two dual ratings with respect to that pair.

Definition 4. Let $R(i, j) = \{\mathbf{r} \in \mathcal{R} : r_i > r_j\}$, a probability space is called rank-differentiable with (i, j) , if for any $\mathbf{r} \in R(i, j)$, $P(\mathbf{r}|\mathbf{x}) \geq P(\mathbf{r}'|\mathbf{x})$, and there exist at least one ratings $\mathbf{r} \in R(i, j)$, s.t. $P(\mathbf{r}|\mathbf{x}) > P(\mathbf{r}'|\mathbf{x})$, where \mathbf{r}' is the dual ratings of \mathbf{r} .

Definition 5. A probability space is called rank-differentiable, if there exists an equivalence class $[\mathbf{r}^*]$, s.t. $\mathcal{P}(\mathbf{r}^*) = \{(i, j) : \text{the probability space is rank-differentiable with } (i, j)\}$, where $\mathcal{P}(\mathbf{r}^*) = \{(i, j) : r_i^* > r_j^*\}$. We will also call this probability space a RDPS or rank-differentiable with $[\mathbf{r}^*]$.

5.2 Consistency with WPD

Following the proof technique in [11], we prove that p-ListMLE is consistent with WPD, as shown in the following theorem.

Theorem 1. We assume that the probability space is rank-differentiable, then the surrogate loss function in p-ListMLE is consistent with WPD.

The proof of the theorem is similar to Theorem 5 in [11], and is based on the following theorem.

Theorem 2. We assume that the probability space is rank-differentiable with an equivalence class $[\mathbf{r}^*]$. let f be a function such that $R_p(f|\mathbf{x}) = \inf_{h \in \mathcal{F}} R_p(h|\mathbf{x})$, then for any object pair $(x_i, x_j), r_i^* > r_j^*$, we have $f(x_i) > f(x_j)$.

Proof. (1) We assume that $f(x_i) < f(x_j)$, and define f' as the function such that $f'(x_i) = f(x_j), f'(x_j) = f(x_i), f'(x_k) = f(x_k), \forall k \neq i, j$. We can then get the

following equation,

$$\begin{aligned} & R_p(f'|\mathbf{x}) - R_p(f|\mathbf{x}) \\ &= \sum_{\substack{\mathbf{r}, \mathbf{r}', \\ \mathbf{r} \in R(i, j)}} \sum_{k: r_j < r_k < r_i} \alpha(r_k) [P(\mathbf{r}|\mathbf{x}) - P(\mathbf{r}'|\mathbf{x})] \\ & \quad \times \log \left(\frac{\sum_{l=y_k, l \neq y_j}^n \exp(f(x_{\mathbf{y}^{-1}(l)})) + \exp(f(x_i))}{\sum_{l=y_k, l \neq y_j}^n \exp(f(x_{\mathbf{y}^{-1}(l)})) + \exp(f(x_j))} \right) \\ &+ \sum_{\substack{\mathbf{r}, \mathbf{r}', \\ \mathbf{r} \in R(i, j)}} [\alpha(r_i) - \alpha(r_j)] [\log(f(x_i)) - \log(f(x_j))] \\ & \quad \times [P(\mathbf{r}|\mathbf{x}) - P(\mathbf{r}'|\mathbf{x})] \\ &+ \sum_{\substack{\mathbf{r}, \mathbf{r}', \\ \mathbf{r} \in R(i, j)}} \alpha(r_j) [P(\mathbf{r}|\mathbf{x}) - P(\mathbf{r}'|\mathbf{x})] \\ & \quad \times \log \left(\frac{\sum_{l=y_j+1}^n \exp(f(x_{\mathbf{y}^{-1}(l)})) + \exp(f(x_i))}{\sum_{l=y_j+1}^n \exp(f(x_{\mathbf{y}^{-1}(l)})) + \exp(f(x_j))} \right), \end{aligned}$$

According to the conditions of RDPS and the requirements of $\alpha(\cdot)$, we can obtain

$$R_p(f'|\mathbf{x}) < R_p(f|\mathbf{x}).$$

This is a contradiction with $R_p(f|\mathbf{x}) = \inf_{h \in \mathcal{F}} R_p(h|\mathbf{x})$. Therefore, we have proven that $f(x_i) \leq f(x_j)$.

(2) Now we assume that $f(x_i) = f(x_j) = f_0$. From the assumption $R_p(f|\mathbf{x}) = \inf_{h \in \mathcal{F}} R_p(h|\mathbf{x})$, we can get

$$\left. \frac{\partial R_\Phi(f|\mathbf{x})}{\partial f(x_i)} \right|_{f_0} = 0, \quad \left. \frac{\partial R_\Phi(f|\mathbf{x})}{\partial f(x_j)} \right|_{f_0} = 0.$$

Accordingly, we can obtain two equations as follows:

$$\sum_{\substack{\mathbf{r}, \mathbf{r}', \\ \mathbf{r} \in R(i, j)}} A_1 P(\mathbf{r}|\mathbf{x}) + A_2 P(\mathbf{r}'|\mathbf{x}) = 0, \quad (12)$$

$$\sum_{\substack{\mathbf{r}, \mathbf{r}', \\ \mathbf{r} \in R(i, j)}} B_1 P(\mathbf{r}|\mathbf{x}) + B_2 P(\mathbf{r}'|\mathbf{x}) = 0, \quad (13)$$

where,

$$\begin{aligned} & A_1 = B_2 \\ &= \sum_{k: r_j < r_i < r_k} \alpha(r_k) \frac{\exp(f_0)}{\sum_{l=y_k}^n \exp(f(x_{\mathbf{y}^{-1}(k)}))} \\ & \quad + \alpha(r_i) \left(-1 + \frac{\exp(f_0)}{\sum_{l=y_i}^n \exp(f(x_{\mathbf{y}^{-1}(k)}))} \right), \end{aligned}$$

$$\begin{aligned}
A_2 &= B_1 \\
&= \sum_{k:r_j < r_i < r_k} \alpha(r_k) \frac{\exp(f_0)}{\sum_{l=y_k}^n \exp(f(x_{\mathbf{y}^{-1}(k)}))} \\
&\quad + \sum_{k:r_i < r_k < r_j} \alpha(r_k) \frac{\exp(f_0)}{\sum_{l=y_k}^n \exp(f(x_{\mathbf{y}^{-1}(k)}))} \\
&\quad + \alpha(r_i) \frac{\exp(f_0)}{\sum_{l=y_i}^n \exp(f(x_{\mathbf{y}^{-1}(l)}))} \\
&\quad + \alpha(r_j) \left(-1 + \frac{\exp(f_0)}{\sum_{l=y_j}^n \exp(f(x_{\mathbf{y}^{-1}(l)}))} \right)
\end{aligned}$$

Based on the requirements of RDPS and $\alpha(\cdot)$, we can obtain that,

$$\begin{aligned}
&\sum_{\substack{\mathbf{r}, \mathbf{r}' \\ \mathbf{r} \in R(i,j)}} (A_1 - B_1)P(\mathbf{r}|\mathbf{x}) + (A_2 - B_2)P(\mathbf{r}'|\mathbf{x}) \\
&= \sum_{\substack{\mathbf{r}, \mathbf{r}' \\ \mathbf{r} \in R(i,j)}} (A_1 - A_2)[P(\mathbf{r}|\mathbf{x}) - P(\mathbf{r}'|\mathbf{x})] \\
&\leq \sum_{\substack{\mathbf{r}, \mathbf{r}' \\ \mathbf{r} \in R(i,j)}} (\alpha(r_j) - \alpha(r_i))[P(\mathbf{r}|\mathbf{x}) - P(\mathbf{r}'|\mathbf{x})] < 0.
\end{aligned}$$

This is a contradiction with Eq. (12). Therefore, we actually have proven that $f(x_i) > f(x_j)$. \square

5.3 Discussion

In [21], ListMLE is proved to be consistent with permutation level 0-1 loss. However, consistency with permutation level 0-1 loss does not mean consistency with NDCG. For example, it has been proven in [18] that losses in RankCosine [17] and ListNet [4] are not consistent with NDCG. Since that we have proven that p-ListMLE is consistent with WPDL, it is natural to study the relationship between WPDL and NDCG. Here we show that WPDL with a certain weight, referred to as *difference-weight pairwise disagreement loss* (DWPD for short), is equivalent to NDCG with a certain discount function, referred to as *sharp-NDCG*. In this way, p-ListMLE is better than ListMLE theoretically.

The formal definition of NDCG is as follows.

$$NDCG@n(f; \mathbf{x}, \mathbf{y}) = \frac{1}{N_n} \sum_{i=1}^n Gain(r(y_i)) Disc(\pi_f(i)),$$

where *Gain* is the gain function which gives larger scores to objects with larger labels and *Disc* is the discount function which gives larger scores to objects ranked higher in π_f . $r(y_i)$ is a function to mapping position y_i into relevance score. For consistency with above formulation, we use r_i to denote $r(y_i)$, and define $r_i = n - y_i$, $\pi_f(i)$ is the position of document x_i in permutation π_f and N_n is a normalization factor.

We define WPDPL, and sharp-NDCG as follows, respectively.

$$\begin{aligned}
l_w(f; \mathbf{x}, \mathbf{y}) &= \sum_{i,j,r_i > r_j} (Gain(r_i) - Gain(r_j)) 1_{\{f(x_i) \leq f(x_j)\}}, \\
sharp-NDCG@n(f; \mathbf{x}, \mathbf{y}) &= \frac{\sum_{i=1}^n (n - \pi(i)) Gain(r_i)}{N'_n},
\end{aligned}$$

where N'_n is a normalization factor.

The following theorem shows the relationship between DWPD and sharp-NDCG.

Theorem 3. We denote $\mathcal{F}_0 = \{f \in \mathcal{F} : \forall \mathbf{x}, f(x_i) \neq f(x_j), \forall i, j\}$, then $\forall (\mathbf{x}, \mathbf{y}), f, g \in \mathcal{F}_0$, we have

$$\begin{aligned}
l_w(f; \mathbf{x}, \mathbf{y}) &< l_w(g; \mathbf{x}, \mathbf{y}) \\
\Leftrightarrow sharp-NDCG@n(\pi_f, \mathbf{x}, \mathbf{y}) &> sharp-NDCG@n(\pi_g, \mathbf{x}, \mathbf{y}).
\end{aligned}$$

Proof. According to the definition of sharp-NDCG, it has the following form:

$$sharp-NDCG@n(\pi, \mathbf{y}) = \sum_{i=1}^n Gain(r_i)(n - \pi(i)).$$

Rewrite $n - \pi(i)$ and $n - \pi_y(i)$ as $\sum_{j=1}^m 1_{\{\pi(i) - \pi(j) < 0\}}$ and $\sum_{j=1}^m 1_{\{\pi_y(i) - \pi_y(j) < 0\}}$, respectively, then the following equation holds,

$$\begin{aligned}
&\sum_{i=1}^n (n - \pi_y) Gain(r_i) - sharp-NDCG@n(\pi; \mathbf{x}, \mathbf{y}) \\
&= \sum_{i=1}^n \sum_{j \neq i} Gain(r(y_i)) (1_{\{\pi_y(i) - \pi_y(j) < 0\}} - 1_{\{\pi(i) - \pi(j) < 0\}}),
\end{aligned}$$

Thus, for each pair $(i, j), r_i > r_j$, the term becomes:

$$\begin{aligned}
&Gain(r_i) 1_{\{\pi(i) - \pi(j) > 0\}} - Gain(r(y_j)) 1_{\{\pi(j) - \pi(i) < 0\}} \\
&= Gain(r_i) 1_{\{\pi(i) - \pi(j) > 0\}} - Gain(j) 1_{\{\pi(i) - \pi(j) > 0\}} \\
&= (Gain(r_i) - Gain(r(y_j))) 1_{\{\pi(i) - \pi(j) > 0\}},
\end{aligned}$$

Further considering the relationship between π and f , the following equation holds:

$$\begin{aligned}
&\sum_{i=1}^m (n - \pi_y) Gain(r_i) - sharp-NDCG@m(\pi; \mathbf{x}, \mathbf{y}) \\
&= l_w(f; \mathbf{x}, \mathbf{y}).
\end{aligned}$$

Since $\sum_{i=1}^m (n - \pi_y) Gain(r_i)$ is just dependent on \mathbf{y} , the result in the theorem has been proved. \square

Note that in applications such as information retrieval, people usually use the following specific discount and gain functions.

$$Gain(i) = 2^{r(y_i)} - 1, Disc(\pi(i)) = \frac{1}{\log_2(1 + \pi(i))}.$$

According to the original paper of NDCG [9], however, this is not the only choice. Actually, the only difference between sharp-NDCG and the above NDCG is that the discount function in sharp-NDCG is sharper.

In summary, we have obtain that:

- (1) p-ListMLE is statistically consistent with WPD, where $D(r_i, r_j) = \alpha(r_j) - \alpha(r_i)$;
- (2) WPD with $D(r_i, r_j) = Gain(r_i) - Gain(r_j)$ is equivalent to sharp-NDCG;
- (3) sharp-NDCG is intrinsic similar with NDCG.

Therefore, we can expect to obtain better performance w.r.t NDCG if we define $\alpha(r_i) = Gain(r_i) = 2^{n-y_i} - 1$.

6 EXPERIMENTS

In this section, we conduct experiments on benchmark data sets LETOR 4.0² to show that p-ListMLE can achieve better performances compared to ListMLE as well as other state-of-the-art listwise learning-to-rank algorithms.

As the setting in this paper is listwise ranking, we choose two datasets in LETOR, i.e. MQ2007-list and MQ2008-list, in which the ground-truth is a full order ranking list. In MQ2007-list, there are about 1700 queries and 700 documents per query on average. In MQ2008-list, there are about 800 queries and 1000 documents per query on average. Both query sets are from Million Query track of TREC 2007 and TREC 2008.

We implement both ListMLE and p-ListMLE by Stochastic Gradient Descent (SGD). In p-ListMLE, we set $\alpha(i)$ as $2^{n-i} - 1$, as guided in the above section. The stopping criteria is chosen from $\{0.1^i\}_{i=1}^6$ to control when to stop, and the learning rates are selected from $\{0.1^i\}_{i=1}^5$ with the maximal number of iterations 500. Evaluation measure NDCG is adopted to evaluate the test performances of both algorithms, where the multi-level ground-truth label is taken as $l(x_i) = n - y_i$, where y_i is the rank of item i in the listwise ground-truth. For empirical comparison, we also include two other state-of-the-art listwise learning-to-rank algorithms such as ListNet and RankCosine, and the implementation is conducted strictly under the standard setting as shown in [4] and [17]. The experimental results are shown in Figure 1.

From the results, we can see that p-ListMLE significantly outperform ListMLE with NDCG as evaluation measure on both datasets. Taking NDCG@10 as an example, the improvement of p-ListMLE over ListMLE is 0.95% and 0.54% on MQ2007-list and MQ2008-list, respectively.

²<http://research.microsoft.com/en-us/um/beijing/projects/letor/>.

Moreover, when comparing p-ListMLE with the other two baseline methods, we can see that it can also outperform traditional listwise ranking methods significantly. We also take NDCG@10 as an example, the improvement of p-ListMLE over ListNet is 0.85% and 0.22% on MQ2007-list and MQ2008-list, respectively. While, the improvement of p-ListMLE over RankCosine is 1.48% and 0.32% on MQ2007-list and MQ2008-list, respectively.

7 CONCLUSION

In this paper, we address the problem of ListMLE that the position importance is highly ignored, which is however very important for ranking. We propose a new listwise ranking algorithm, namely position-aware ListMLE (p-ListMLE) to amend the problem. In p-ListMLE, ranking is viewed as a sequential learning process, with each step learning a subset of parameters which minimize the corresponding stepwise probability distribution. We prove that p-ListMLE is consistent with WPD, which is equivalent to a certain form of NDCG. In addition, our experimental results on benchmark datasets show that p-ListMLE can significantly outperform ListMLE. Therefore, we have demonstrated both theoretically and empirically that p-ListMLE is better than ListMLE.

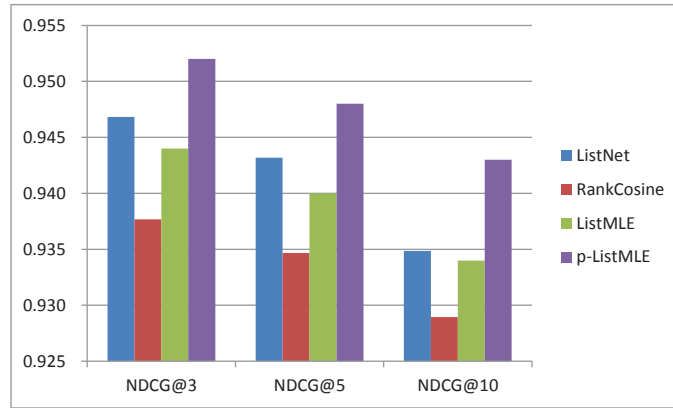
For the future work, we plan to further investigate the problem of how to set the position factor $\alpha(\cdot)$ in practice, or how to guide the settings from other theoretical aspects.

Acknowledgments

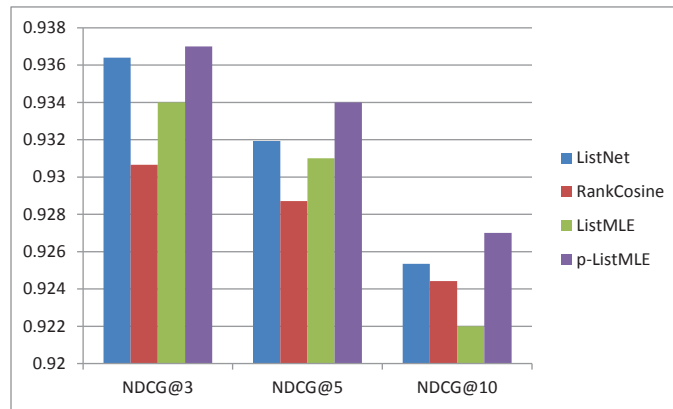
This research work was funded by the 973 Program of China under Grants No. 2012CB316303 and No. 2014CB340401, the 863 Program of China under Grants No. 2012AA011003, National Natural Science Foundation of China under Grant No. 61232010 and No. 61203298.

References

- [1] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proceedings of the 22th International Conference on Machine Learning (ICML 2005)*, pages 89–96, 2005.
- [2] C. Burkley and E. M. Voorhees. *Retrieval System Evaluation*. MIT Press, 2005.
- [3] Y. Cao, J. Xu, T.-Y. Liu, H. Li, Y. Huang, and H.-W. Hon. Adapting ranking svm to document retrieval. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '06*, pages 186–193, New York, NY, USA, 2006. ACM.



(a) MQ2007-list



(b) MQ2008-list

Figure 1: Performance Comparison Between p-ListMLE and ListMLE

- [4] Z. Cao, T. Qin, T. Y. Liu, M. F. Tsai, and H. Li. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th International Conference on Machine Learning (ICML 2007)*, pages 129–136, 2007.
- [5] O. Chapelle, D. Metzler, Y. Zhang, and P. Grinspan. Expected reciprocal rank for graded relevance. In *Proceedings of the 18th ACM conference on Information and knowledge management, CIKM '09*, pages 621–630, New York, NY, USA, 2009. ACM.
- [6] W. Chen, T. yan Liu, Y. Lan, Z. Ma, and H. Li. Ranking measures and loss functions in learning to rank. In *Twenty-Fourth Annual Conference on Neural Information Processing Systems*, pages 315–323, 2009.
- [7] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969, 2003.
- [8] C.-L. Hwang and A. S. M. Masud. *Multiple Objective Decision Making, Methods and Applications*. springer-Verlag, 1979.
- [9] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Informations and Systems*, 20(4):422–446, 2002.

- [10] T. Joachims. Optimizing search engines using click-through data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2002)*, pages 133–142, 2002.
- [11] Y. Lan, J. Guo, X. Cheng, and T.-Y. Liu. Statistical consistency of ranking methods in a rank-differentiable probability space. In P. Bartlett, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1241–1249, 2012.
- [12] Y. Lan, T.-Y. Liu, Z. Ma, and H. Li. Generalization analysis of listwise learning-to-rank algorithms. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09*, pages 577–584, New York, NY, USA, 2009. ACM.
- [13] Y. Lan, S. Niu, J. Guo, and X. Cheng. Is top-k sufficient for ranking? In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management, CIKM '13*, pages 1261–1270, New York, NY, USA, 2013. ACM.
- [14] P. Li, C. Burges, and Q. Wu. Mcrank: Learning to rank using multiple classification and gradient boosting. In *Twenty-Second Annual Conference on Neural Information Processing Systems*, 2007.
- [15] T.-Y. Liu. Learning to rank for information retrieval. *Foundation and Trends on Information Retrieval*, 3:225–331, 2009.
- [16] S. Niu, J. Guo, Y. Lan, and X. Cheng. Top-k learning to rank: labeling, ranking and evaluation. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval, SIGIR '12*, pages 751–760, New York, NY, USA, 2012. ACM.
- [17] T. Qin, X.-D. Zhang, M.-F. Tsai, D.-S. Wang, T.-Y. Liu, and H. Li. Query-level loss functions for information retrieval. *Information Processing & Management*, 2007.
- [18] P. Ravikumar, A. Tewari, and E. Yang. On ndcg consistency of listwise ranking methods. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, volume 15 of *JMLR Workshop and Conference Proceedings*, pages 618–626, 2011.
- [19] F. Xia, T.-Y. Liu, and H. Li. Statistical consistency of top-k ranking. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 2098–2106. 2009.
- [20] F. Xia, T.-Y. Liu, J. Wang, W. Zhang, and H. Li. Listwise approach to learning to rank: theory and algorithm. In *Proceedings of the 25th international conference on Machine learning, ICML '08*, pages 1192–1199, New York, NY, USA, 2008. ACM.
- [21] F. Xia, T. Y. Liu, J. Wang, W. S. Zhang, and H. Li. Listwise approach to learning to rank - theory and algorithm. In *Proceedings of the 25th International Conference on Machine Learning (ICML 2008)*, 2008.
- [22] J. Xu and H. Li. Adarank: A boosting algorithm for information retrieval. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '07*, pages 391–398, New York, NY, USA, 2007. ACM.

Continuously indexed Potts models on unoriented graphs

Landrieu Loïc † *

† Inria - Sierra Project-Team
École Normale Supérieure
Paris, France

Guillaume Obozinski *

* Université Paris-Est, LIGM
École des Ponts - ParisTech
Marne-la-Vallée, France

Abstract

This paper introduces an extension to undirected graphical models of the classical *continuous time* Markov chains. This model can be used to solve a transductive or unsupervised multi-class classification problem at each point of a network defined as a set of nodes connected by segments of different lengths. The classification is performed not only at the nodes, but at every point of the edge connecting two nodes. This is achieved by constructing a *Potts process* indexed by the continuum of points forming the edges of the graph.

We propose a homogeneous parameterization which satisfies Kolmogorov consistency, and show that classical inference and learning algorithms can be applied.

We then apply our model to a problem from geomatics, namely that of labelling city blocks automatically with a simple typology of classes (e.g. collective housing) from simple properties of the shape and sizes of buildings of the blocks. Our experiments shows that our model outperform standard MRFs and a discriminative model like logistic regression.

1 INTRODUCTION

Connections in networks typically have a length or weight that gives a measure of distance between the nodes connected, or the intensity of their interaction. This length information has been used to perform unsupervised or semi-supervised classification on graphs based among others on graph partitioning algorithms (see e.g. Zhu and Goldberg, 2009). When defining probabilistic graphical models on such networks, it is not clear how to take this distance into account naturally so that the interaction decreases with the distance. In this paper, we propose an unoriented counterpart of the continuous-time Markov process on a tree proposed by Holmes and Rubin (2002) which is naturally generalized to any unoriented graph.

In a continuous time Markov chain, a random state X_t is associated with every point $t \in \mathbb{R}_+$. The generalization to a continuous tree mode considered by Holmes and Rubin (2002) is most simply described through its application in phylogenetics. The phylogenetic tree of a family of species is assumed given as a directed tree with branches of different lengths. The length of the branches measure the genetic distance between extant or extinct species. Branching nodes are associated with speciation events. Each point of each branch of the tree corresponds to the form taken by a species as it existed at one time in the past and the variable modeled as a random process and defined at each such point is typically a discrete trait of that species such as the nucleic acid among $\{A, C, T, G\}$ at a certain position in the DNA. In the absence of speciation event, the state evolves like a continuous-time Markov chain, with time here being measure in terms of the genetic distance along an edge. When a branching occurs, the Markov chain is split into two identical states which continue to evolve independently. For that process, if the edges of the trees are identified with line segments, there is a random variable X_t associated with every point t of each of these segments. Since a tree is simply connected, removing point t will split the tree in at least two components, and with this model we have the fundamental Markov property that the subprocesses defined on each component are conditionally independent given X_t .

We aim to extend these models in two ways. First, these continuously indexed processes are fundamentally oriented. This stems for the fact that the continuous Markov chain in this model is *homogeneous*, which implies that the conditional distributions forward in time are constant, a property which, while true forward, is not in general true backwards in time. This implies in particular that all marginals of the process on any finite set of points including at least all nodes of degree different than two is naturally parameterized as a product of conditionals $p(x_s|x_t)$ whose value depends on the graph only through the distance between s and t . We aim to propose natural parameterization for *unoriented continuously indexed models* with the same Markov property as the oriented trees. Second,

the considered models are simply connected and we would like to propose an extension from weighted trees to general weighted graphs, where all edges are identified with a real segment of lengths equal to their weights, and which satisfy the Markov property in the sense that if a finite set of points A on these segments cuts the graph into several connected components, the processes on the two subgraphs are conditionally independent given $(X_a)_{a \in A}$. The obtained models will be Potts models that take into account in a natural way the length of the edges and such that the interaction between two nodes decreases with the distance separating them.

After a discussion of related work, we first consider the simplest case of an unoriented continuous chain for which we propose an exponential family parametrization. Next, we show how this parametrization is naturally extended to general unoriented continuous graphs. We derive the marginal log-likelihood of different subsets of nodes, as well as the form of its gradients, and show that inference and learning in these models can be obtained with classical algorithms. We then extend the model and algorithms to the hidden Markov random field case where a feature vector is attached to a certain number node. In terms of experiments, we consider first a transductive classification problem from geomatics, which consists in assigning city blocks to different classes from simple buildings characteristics, while taking into account the distances between the blocks. Then we illustrate the possibility of using the model for transfer learning in order to refine predictions for city blocks from a new entirely unlabelled city.

2 RELATED WORK

The model we consider in this work can be viewed as an extension to undirected graphs of the continuous time Markov chain (CTMC). The continuous-time Markov chain (Norris, 1997) is a fundamental model in probability and statistics for random variables that take values in a set of discrete states and that can transition at any point in continuous time from one state to another. Beyond its theoretical value, it has been applied directly in queuing theory, for the statistical modeling of chemical reactions and in genetics.

In genetics, CTMC models have been notably used to propose models of the evolution of DNA at the nucleotide level (Nielsen, 2005; Durrett, 2008), with among several others, the celebrated Jukes-Cantor model. In this context, these models have been extended to directed trees, where the tree corresponds to a phylogeny of species or of proteins, and which has been used to estimate rate matrices or for genetic sequence alignment (Von Bing and Speed, 2004). Like for CTMCs, the fact that these models are continuous arise from temporality, and the models derived are thus intrinsically oriented. For these CTMC on trees, Holmes and Rubin (2002) proposed an exponential family parametrization of the likelihood and showed that it was possible to

design an EM algorithm to learn the rate matrices modeling the substitution of DNA bases over time, in a way that generalizes the classical EM algorithm on trees.

As is the case for the CTMC, continuously indexed processes arise typically as the limit of discretely indexed processes. Along these lines, Yaple and Abrams (2013) consider a continuum limit of the Ising model on a regular grid where the lengths of the edges are infinitesimal and use it to characterize the patterns of magnetic polarity in ferromagnetic materials through the resolution of integro-differential equations.

A different but also recent line of research combining ideas from the graphical models literature with stochastic processes is known under the name of *continuous time Bayesian networks* (CTBNs, Nodelman et al., 2002). These are models of structured multivariate stochastic processes in time in which the interaction between the different components of the process can be modeled by a graphical model. These models are quite different than the continuous time tree models or the models we will propose in this paper in that, for CTBNs, the graphical model structure is somehow orthogonal to the direction of time which is the unique global oriented continuous variable for the process.

Last but not least, a common family of approaches which take into account the length of edges in a graph in the context of unsupervised or semi-supervised classification are the graph partitioning and related spectral clustering techniques (see e.g. Zhu and Goldberg, 2009, chap. 5). A review of these techniques is beyond the scope of this paper. We however discuss how these methods differ and are not directly comparable to ours in section 7.

3 NOTATIONS

All multinomial variables considered in the paper take values in $\mathcal{K} = \{1, \dots, K\}$ and are represented by the indicator vector $x \in \{0, 1\}^K$ whose sole non zero entry is x_k when the multinomial is the k th state. We thus define $\mathcal{X} = \{x \in \{0, 1\}^K \mid \sum_{k \in \mathcal{K}} x_k = 1\}$. Given a vector $x \in \mathbb{R}^K$, $\text{Diag}(x)$ is the diagonal matrix whose elements are the entries in x .

We use \odot (resp. \oslash) to denote the Hadamard product (resp. division), that is the entrywise multiplication (resp. division) of matrices.

We will denote nodes of graphical model with the sans-serif font a, b , and set of nodes with upper capitals of the same font: A, B .

4 CONTINUOUS GRAPH POTTS MODELS

4.1 An unoriented continuous chain model

To derive a parameterization of the model, we start with the case of an unoriented chain that we identify with the $[0, l]$

segment, where without loss of generality l is an integer. We will denote by X_a a multinomial random variable associated with the point $a \in [0, l]$. Before defining the process at any point of the segment, we model the joint distribution of the random variables X_k for k an integer in $\{0, \dots, l\}$. Denoting by $x_k \in \{0, 1\}^K$ an instance of X_k , and assuming that both unary and binary potentials are constant, the joint distribution of $(X_k)_{k \in \{0, 1, \dots, l\}}$ can be written in multiplicative form as

$$p(x_0, x_1, \dots, x_l; U, h) \propto \prod_{k=0}^l h^\top x_k \prod_{k=0}^{l-1} x_k^\top U x_{k+1},$$

with $h \in \mathbb{R}_{+*}^K$ the vector of unary potential values and $U \in \mathbb{R}_{+*}^{K \times K}$ the matrix of binary potential values. For reasons of symmetry and invariance along the chain, we assume that those parameters do not depend on the position k and that $U = U^\top$. Note that, while similar in spirit, the assumption that these parameters are constant is different from assuming that the Markov chain is homogeneous; we discuss this point in section 7. To get concise forms for the distributions induced on subsets of the X_k s by marginalization, we introduce further $H = \text{Diag}(h)$ and $W = H^{\frac{1}{2}} U H^{\frac{1}{2}}$. If in particular we marginalize all variables except for the extreme points of the segment we then have

$$\begin{aligned} p(x_0, x_l; W, h) &\propto \sum_{x_1 \dots x_{l-1}} \prod_{i=0}^{l-1} x_i^\top U x_{i+1} \prod_{i=0}^l h^\top x_i \\ &\propto h^\top x_0 \left(x_0^\top H^{-\frac{1}{2}} W^l H^{-\frac{1}{2}} x_l \right) h^\top x_l, \end{aligned}$$

(See appendix for details).

Similar calculations show that, for any sequence $a_0 = 0 < a_1 < \dots < a_m = l$ with $a_k \in \{0, \dots, l\}$, denoting $d_j = d(a_j, a_{j-1}) = a_j - a_{j-1}$ the distances between consecutive nodes and $A = \{a_0, \dots, a_m\}$, we have:

$$p(x_A; W, h) \propto \prod_{j=0}^m h^\top x_{a_j} \prod_{j=1}^m x_{a_{j-1}}^\top H^{-\frac{1}{2}} W^{d_j} H^{-\frac{1}{2}} x_{a_j}.$$

By simply taking the logarithm of this expression we obtain a curved exponential family of distributions with log-likelihood

$$\ell(x_A; \theta) = \sum_{j=0}^m \eta^\top x_{a_j} + \sum_{j=0}^{m-1} x_{a_j}^\top \Lambda(\theta, d_j) x_{a_{j+1}} - A(\theta), \quad (4.1)$$

with $\forall k \in \mathcal{K}$, $\eta_k = \log(h_k)$, $\theta = (W, \eta)$, A the log-partition function and where $\Lambda(\theta, d)$ is defined entrywise by $[\Lambda(\theta, d)]_{kk'} = \log([H^{-\frac{1}{2}} W^d H^{-\frac{1}{2}}]_{kk'})$.

It is now very natural to try and use this formula to extend the definition of the process to any sequence of points $a_0 = 0 < a_1 < \dots < a_m = l$ that are no longer restricted to take integer values. This requires however that for all

for all $s \geq 0$, W^s should be a well defined real valued matrix with non-negative (or for learning purposes positive) entries. The fact that W is real symmetric and that all its powers should be real implies that it should have non-negative eigenvalues. Since we can approximate a low rank matrix with a full rank matrix, we assume for convenience that all its eigenvalues are positive (any low rank matrix can be approximated by a full rank one). W is then a matrix exponential $W = \exp(\Pi)$. The fact that all its powers should have non-negative entries implies in particular that for any s , W^s is *completely positive*¹. We therefore need to characterize which conditions on Π are needed to obtain a valid W . Note that Π can be viewed as the counterpart of the rate matrix for CTMCs.

4.2 Infinitesimal generator Π

To easily compute the matrix exponential we use the eigen-decomposition of Π :

$$\Pi = P^\top \Sigma P, \quad \Sigma = \text{Diag}(\sigma), \quad P^\top P = P P^\top = I_K \quad (4.2)$$

and exponentiate its eigenspectrum².

In the context of learning, it is natural to assume that the entries of W^s are actually strictly positive so that the log-likelihood is always finite. The following lemma provides sufficient and necessary conditions on Π for the entries of $\exp(l\Pi)$ to be either non negative or positive.

Lemma 1. *For Π a square matrix, $[\exp(l\Pi)]_{i,j} \geq 0 \forall l \in \mathbb{R}_+$ and $\forall i, j$ if and only if $\Pi_{i,j} \geq 0$ for all $i \neq j$. Similarly, $[\exp(l\Pi)]_{i,j} > 0$ for all i, j and $\forall l \in \mathbb{R}_+^*$, if and only if the sequences $(u_{i,j}^{(k)})_{k \in \mathcal{N}}$ with $u_{i,j}^{(k)} = [\Pi^k]_{i,j}$ is such that its first non-zero value exists and is strictly positive, for all $i \neq j$.*

This lemma is proved in the appendix.

It is easy to see from the proof of the lemma that $\Pi_{i,j} > 0$ for $i \neq j$ is a sufficient condition for $[\exp(l\Pi)]_{i,j}$ to be positive for all i, j and for all $l \in \mathbb{R}_+^*$.

Note that the likelihood obtained in (4.1) is invariant by a multiplication of H or U and thus of W by a positive scalar, because of normalization. As a result it is also invariant by addition of a constant multiple of the identity matrix to Π or equivalently to σ .

This means that the likelihood is invariant by addition of an arbitrary identical constant to all the eigenvalues $(\sigma_i)_{i \in \mathcal{K}}$. In particular, it is possible to choose this constant sufficient large to guarantee that the diagonal of Π is positive. This implies that it will be conveniently possible to parameterize the model by the entrywise logarithm of Π .

¹ $A \in \mathbb{R}^{K \times K}$ is *completely positive* iff there exists $B \in \mathbb{R}_+^{K \times m}$ with $A = B B^\top$ (see e.g. Seber (2008) p. 223).

²One caveat of this parametrization is that if W is close to low rank, the corresponding eigenvalues in σ have to take large negative values. This could be addressed by working with $(\sigma_k^{-1})_{k \in \mathcal{K}}$.

4.3 Existence of the process on the chain

Proposition 2. *There exists a stochastic process $(X_a)_{a \in [0, l]}$ defined at all points of the segment $[0, l]$ whose finite marginal on any finite set of points containing a_0 and a_l is given by (4.1).*

Proof. Let $A = \{a_0, \dots, a_m\}$ and $B = \{b_0, \dots, b_n\}$ two such sets with $a_0 = b_0 = 0$ and $a_m = b_n = l$. It is clear that using (4.1) to define a joint probability distribution on $(X_a)_{a \in A \cup B}$, the distribution obtained by marginalization of elements of $A \setminus B$ using the same type of derivation used in (4.1) is still of the form of (4.1). Since the same holds for $B \setminus A$, we just showed that the collection of proposed marginals are consistent and by Kolmogorov's extension theorem (Chung and Speyer, 1998, chap. 6). This proves the existence of the process. \square

4.4 Extending the model to graphs

4.4.1 Real graphs

To extend the model we proposed on a segment to undirected trees and more generally to undirected graphs, we first define what we will call *continuous graphs* or *real graphs*³. Given a weighted graph $G = (V, E)$ with the weight d_{ab} associated with the edge $(a, b) \in E$, we define the associated *real graph* \mathcal{G} as the space constructed as the union of line segments of lengths d_{ab} associated with the edges $(a, b) \in E$ and whose extreme points are respectively identified with the nodes a and b through an equivalence relation. Put informally, a real graph is the set of line segments that we usually draw to represent an abstract graph. For any pair of points a', b' on the same segment $[a, b]$, we will denote by $d_{a'b'}$ the length of that subsegment.

It should be noted that, in a real graph, the segments connecting a node of degree two are essentially merged into a single segment by concatenation. We will call all nodes of degree different than two *junction nodes*. Conversely, identifying nodes and points in the real graph, any point that is not a junction node can actually be viewed as a degree two node.

Definition 3. *Let S be the set of junction nodes. Given A a set of points on the real graph, we will call the induced discrete graph on $A \cup S$, denoted by G_A the graph with vertices $A \cup S$ and whose edges E_A link the nodes that can be joined on the real graph by segments not containing elements of $A \cup S$: $E_A = \{(a, b) \mid [a, b] \cap (A \cup S) = \emptyset\}$. To distinguish them from $S \setminus A$, we will call the set of nodes in A observed nodes.*

³Real graphs extend the notion of real trees which have been introduced previously in the literature (Chiswell, 2001) and are of interest notably in mathematical cladistics and to construct Brownian trees.

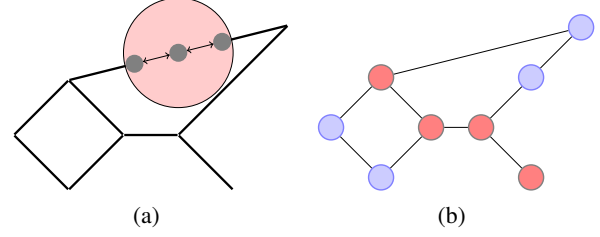


Figure 4.1: (a) Representation of a real graph with a zoom that shows that edges are actually a continuum of nodes linked by infinitesimal unoriented edges. (b) The induced discrete graph associated with the junction nodes in red and the observed nodes in blue.

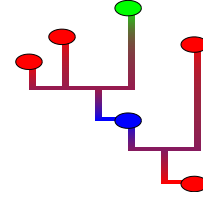


Figure 4.2: (left) Toy example illustrating that the process is defined at all points of the continuous graph. For a model on three classes (red, green blue) each point of each edge is colored with the mixture of these three colors corresponding to the probability of observing each of the classes, given that all the circle nodes are observed with the given colors.

The concepts of real graph, junction node, observed node and induced graph are illustrated on Figure 4.1.

4.4.2 Towards a Potts model on real graphs

To extend the stochastic process previously defined to real graphs, we first define its marginals. In particular, given a set of points $A = \{a_0, \dots, a_m\}$, the marginal on $A \cup S$ is naturally defined as follows: let $G_A = (A \cup S, E_A)$ be the induced discrete graph on $A \cup S$, we propose to define the log-marginal distribution on $(X_a)_{a \in A \cup S}$ as

$$\ell(x_{A \cup S}; \theta) = \sum_{a \in A \cup S} \eta^T x_a + \sum_{(a, b) \in E_A} x_a^T \Lambda(\theta, d_{ab}) x_b - A(\theta), \quad (4.3)$$

with $\theta = (\eta, W)$ which we reparametrize from now on with $\theta = (\eta, \Pi)$. If A does not contain S , then $p(x_A)$ is obtained by marginalizing $x_{S \setminus A}$ out in $p(x_{A \cup S})$.

4.4.3 Existence of the process on a real graph

The existence of the process on a real graph is again proven using Kolmogorov's theorem:

Proposition 4. *There exists a stochastic process $(X_a)_{a \in \mathcal{G}}$ defined at all points of the real graph \mathcal{G} with log-marginals on any set of nodes A containing the junction nodes given by Eq. (4.3).*

Proof. Let A and B be two subsets of nodes on the real graph \mathcal{G} , for which the distributions x_A and x_B are obtained

by marginalizing S out of $x_{A \cup S}$ and $x_{B \cup S}$ in Eq. (4.3). We note that a node on an edge is conditionally independent of any node on a different edge given x_S . Proposition 2 tells us that the marginals are consistent on each edge with fixed endpoints, from which we can deduce that the definition of the definition of the process on $A \cup S$ and $B \cup S$ provided in Eq. (4.3) is consistent since it is obtained by marginalization of the joint distribution at the nodes $A \cup B \cup S$. The process being consistent on A and $A \cup S$ by definition of $p(x_A)$, and similarly on B and $B \cup S$, we have proved Kolmogorov consistency between A and B which in turn proves the existence of the process on the real graph. \square

We will refer to the obtained process, illustrated on Figure 4.2, as a continuous graph Potts model or continuous graph Markov random field (CGMRF).

5 INFERENCE

Probabilistic inference is an operation which is key to learning and making predictions in graphical models. In the case of our continuous graph \mathcal{G} , if we consider any segment $[a, b]$ with $a, b \in S$ and any $a', b' \in [a, b]$, it should be noted that $p(x_{\{a, a', b', b\}}) = p(x_{\{a', b'\}} | x_{\{a, b\}}) p(x_{\{a, b\}})$ where $p(x_{\{a, b\}})$ is computed as a clique marginal of $p(x_S)$, and $p(x_{\{a', b'\}} | x_{\{a, b\}})$ has a simple analytical expression given that reduces to the model on the segment. This implies that marginal distributions on any finite collection of nodes on the same edge can be computed efficiently provided the edge marginals of the induced model on S can be computed efficiently. In spite of the fact that the graph has uncountably many nodes, inference can thus be performed by any classical inference algorithm, i.e. the sum-product algorithm if the graph is a tree and typically approximate inference techniques otherwise, such as loopy belief propagation.

6 LEARNING

In this section, we focus on learning the model from data. Since the process values are only observed at a finite number of points, we are somehow always in the situation where some nodes are unobserved. However, when all junctions nodes are observed the joint likelihood of a given set of nodes has the closed form expression of Eq. (4.3). Since this is a curved exponential family, the log-likelihood is in general not a concave function of the parameters⁴.

To avoid having to cope with positivity constraints, and given the rapid divergence of the likelihood on the boundary of the domain we parameterize the likelihood by η and the entrywise logarithm of Π , since given the remark following lemma 1, it is possible to take Π positive entrywise.

⁴It is however clearly concave when all edges are of the same length, because the constraint of equality of the parameters for all potentials is a convex constraint.

For the CTMC directed tree, Holmes and Rubin (2002) consider the likelihood of the entire process, show that it has a canonical exponential family form with a small number of sufficient statistics and derive an EM algorithm based on this representation to learn the parameters. A similar exponential family form can be obtained for our process, with also a small number of sufficient statistics and in theory it is possible to construct a similar EM algorithm. Unfortunately, in our case the M-step of the algorithm would still require solving a convex optimization problem whose solution is not closed form. We therefore do not pursue further this approach or detail the corresponding canonical exponential family form of the process. We propose instead to optimize the likelihood using a gradient based method. We show that the gradient can be computed from the moments obtained by performing the probabilistic inference on the model in different settings. In the next sections (sections 6.1 - 6.4), we derive the form of the gradient of the likelihood, first when all junction nodes are observed, then, when any set of nodes is observed, and finally, when some nodes are observed and another (typically larger) set of nodes emits observed vectors of features that are each conditionally independent given the state of associated node, as in a hidden Markov random field setting. Since computing the inference is typically intractable in graphs, we introduce a variational approximation in 6.5 that allows for faster (linear) computation. The proofs of lemmas and propositions presented can be found in the appendix.

6.1 Gradient of the likelihood on a segment

Given that the model is parameterized by exponentials of Π , the gradients involve the differential of the matrix exponential. We will therefore repeatedly use the function $\psi_{l, \Pi}$ with $\psi_{l, \Pi}(X) = P^\top ((PXP^\top) \odot \Gamma_{l, \Pi}) P$, where $\Pi = P \text{Diag}(\sigma) P^\top$ is the eigenvalue decomposition of Π and

$$[\Gamma_{l, \Pi}]_{i, j} = \begin{cases} \frac{\exp(l\sigma_i) - \exp(l\sigma_j)}{\sigma_i - \sigma_j} & \text{if } \sigma_i \neq \sigma_j \\ l \exp(l\sigma_j) & \text{if } \sigma_i = \sigma_j. \end{cases}$$

The function ψ is such that the gradient of $x^\top \exp(l\Pi) y$ is $\psi_{l, \Pi}(xy^\top)$. It is essentially switching to the spectral space of Π , where the gradient has a simple multiplicative form given by Γ and then maps the result back to the original space. With this function, we thus have

Lemma 5. *The gradient with respect to variable Π of the log-likelihood ℓ of x_a and x_b on a segment of length l whose end points are a and b can be written as*

$$\nabla_\Pi \ell(x_a, x_b; \theta) = \psi_{l, \Pi}((x_a x_b^\top - \mathbb{E}[X_a X_b^\top]) \odot W^l).$$

6.2 Gradient of the likelihood in a real graph

We now compute the gradient of the log-likelihood for the joint distribution of the nodes $(x_a)_{a \in A}$, with the subset A

containing the junction nodes S . We still denote $G_A = (A, E_A)$ the *induced discrete graph* of A on \mathcal{G} . And since Π is identical for every edge, a direct application of the chain rule implies that:

Proposition 6. *The gradient of the likelihoods are computed⁵ as*

$$\begin{aligned}\nabla_{\Pi} \ell(x_A; \theta) &= \sum_{(a,b) \in E_A} \psi_{d_{ab}, \Pi}((x_a x_b^T - \mu_{ab}) \odot W^{d_{ab}}) \\ \nabla_{\eta} \ell(x_A; \theta) &= \sum_{a \in A} (x_a - \mu_a) - \frac{1}{2} \sum_{(a,b) \in E_A} (x_a - \mu_a + x_b - \mu_b).\end{aligned}$$

with $\mu_a = \mathbb{E}[X_a]$ and $\mu_{ab} = \mathbb{E}[X_a X_b^T]$.

6.3 Partially observed junction nodes

To learn from partially labelled data it is necessary to consider the likelihood of X_B for B a set of nodes that does not necessarily contain S . Let B be a set of observed nodes, i.e. for which we know the states x_B , and A a set of unobserved nodes containing $S \setminus B$. We have the following distributions

$$\begin{aligned}\ell(x_{A \cup B}; \theta) &= \sum_{a \in A \cup B} \eta^T x_a + \sum_{(a,b) \in E_{A \cup B}} x_a^T \Lambda(\theta, d_{ab}) x_b - A_{A \cup B}(\theta) \\ \ell(x_A | x_B; \theta) &= \sum_{a \in A \cup B} \eta^T x_a + \sum_{(a,b) \in E_{A \cup B}} x_a^T \Lambda(\theta, d_{ab}) x_b - A_{A|B}(\theta, x_B)\end{aligned}$$

We can rewrite the log-likelihood as follows (Wainwright and Jordan, 2008) :

$$\ell(x_B; \theta) = A_{A|B}(\Pi, h, x_B) - A_{A \cup B}(\Pi, h),$$

and its gradient are therefore computed as

Proposition 7.

$$\begin{aligned}\nabla_{\Pi} \ell(x_B; \theta) &= \sum_{(a,b) \in E_{A \cup B}} \psi_{d_{ab}, \Pi}((\mu_{ab|B} - \mu_{ab}) \odot W^{d_{ab}}) \\ \nabla_{\eta} \ell(x_B; \theta) &= \sum_{a \in A \cup B} \mu_{a|B} - \mu_a \\ &\quad - \frac{1}{2} \sum_{(a,b) \in E_{A \cup B}} (\mu_{a|B} - \mu_a + \mu_{b|B} - \mu_b).\end{aligned}$$

with $\mu_{ab|B} = \mathbb{E}[X_a X_b^T | X_B = x_B]$, $\mu_{a|B} = \mathbb{E}[X_a | X_B = x_B]$.

6.4 Hidden Markov model

We consider a hidden Markov random field variant of our model in which some nodes have, in addition to the state variable, a feature vector with a state specific distribution. More precisely, we envision to learn from data on a graph in which the states of a set of nodes B are observed and in

which each node in a set A (with $A \cap B \neq \emptyset$) provides an observed feature vectors y_a which is conditionally independent of the rest of the graph given the corresponding node state x_a . For simplicity, we assume that $S \subset A \cup B$.

The joint and conditional distribution of observed and unobserved variables are very similar as above

$$\begin{aligned}\ell(x_{A \cup B}, y_A; \theta, \kappa) &= \sum_{a \in A \cup B} \eta^T x_a + \sum_{a \in A} \log(p(y_a | x_a; \kappa)) \\ &\quad + \sum_{(a,b) \in E_{A \cup B}} x_a^T \Lambda(\theta, d_{ab}) x_b - A_{A \cup B}(\theta, \kappa) \\ \ell(x_A | y_A, x_B; \theta, \kappa) &= \sum_{a \in A \cup B} \eta^T x_a + \sum_{a \in A} \log(p(y_a | x_a; \kappa)) \\ &\quad + \sum_{(a,b) \in E_{A \cup B}} x_a^T \Lambda(\theta, d_{ab}) x_b - A_{A|B}(\theta, \kappa, x_B, y_A),\end{aligned}$$

which allows us to rewrite the likelihood of observations as $\ell(x_B, y_A) = A_{A|B}(\theta, \kappa, y_A, x_B) - A_{A \cup B}(\theta, \kappa)$.

Given that the model for $p(y_a | x_a)$ is Gaussian or at least an exponential family, when envisioning an EM algorithm to learn κ and θ , it is easy to see that the update for κ is closed form while that of θ is not. This motivates a variant of the EM algorithm which does not attempt to maximize with respect to both κ and θ simultaneously but which either maximizes the expected likelihood with respect to κ or maximizes it with respect to θ . The algorithm can then be summarized as an E-M1-E-M2 algorithm, where the E-step is the usual computation of expected sufficient statistics given current parameters, M1 solves for κ in closed form and M2 maximizes with respect to θ using gradient ascent⁶.

6.5 Variational approximation

For graphs with cycles, since inference is intractable, we replace the likelihood by a pseudo-likelihood obtained using a variational approximation of the log-partition. Our variational approximation is the one associated with the entropy of Bethe (see, e.g. section 4.1 in Wainwright and Jordan, 2008), but other choices would be possible. The main motivation behind this approximation is that the exact gradient of this pseudo-likelihood is directly obtained from the pseudo-moments given by loopy BP. In practice, damping needs to be used (see Wainwright and Jordan, 2008, chap. 7).

In term of complexity, the parametrization of CGMRF could suggest that inference is slower than in the discrete setting since the computation of the SVD of Π is required. However, since the number of states is typically much smaller than the number of nodes in the graph, the computational cost of the SVD is negligible compared to

⁵Note that a single spectral decomposition of Π allows to compute $W^{d_{ab}}$ efficiently for all pairs (a, b) .

⁶Note that gradient ascent itself requires to perform some inference to recompute the log-partition function

the overall cost of the algorithm. Hence, inference in the CGMRF is just as hard as for any discrete MRF.

The log-likelihood is a curved exponential family and is in particular not a convex function of the parameters, while it is convex for a standard MRF. As a consequence the pseudo log-likelihood based on the variational approximation is also non-convex. We use gradient descent with a line-search based on the Wolfe conditions to find a local minimum (see Nocedal and Wright, 1999, chap. 3). Empirically the algorithm is not trapped in bad local minima but takes more iterations to converge than the MRF counterpart. Experiments showed that the training for CGMRFs was only two times longer than for regular MRFs.

7 DISCUSSION

In this section, we discuss more precisely features of CGMRFs that are unique or common with other models and approaches existing in the literature.

First, we note that for a tree, our model is not equivalent to that of Holmes and Rubin (2002). Their model uses a constant rate matrix (i.e. the Markov process is homogeneous) while we use constant infinitesimal potentials, which do not lead to a constant rate matrix on any orientation of the tree. If the tree is just the segment $[0, L]$, for s and t with $0 < s < t < L$ a CTMC is such that $p(x_t|x_s)$ only depends on $t - s$ and not on L . By contrast for our model $\log p(x_t|x_s)$ depends also on $L - t$ and $L - s$ since $\log p(x_t|x_s) = x_s^T \Lambda (t-s) x_t + x_t^T \eta + x_t^T \Lambda (L-t) \mathbf{1} - x_s^T \Lambda (L-s) \mathbf{1}$, where for simplicity we omitted the dependance in θ , and $\mathbf{1}$ is the constant vector equal to 1. See the appendix for an illustration and further discussion of the differences between the models.

Our model has in common with graph partitioning techniques and spectral clustering (SC) that the distance between nodes are taken into account. But there are several important differences: first, in SC, there is no model learning in the sense that no parameters are learned to optimize the model (Bach and Jordan (2006) who learn the metric for SC, are an exception). Second, our model captures that there could be different transition probabilities between different classes along the graph which is not possible in SC. Then, the main assumption in SC is that classes are separated by edges of smaller weights so that each class is as disconnected as possible. By contrast, our model authorizes (to some extent) transitions between classes on short edges and moreover permits that each class corresponds to several connected components. Our models extends naturally to a hidden Markov model that makes it possible to include feature vectors for some nodes and not for others, which is not possible with SC techniques.

Another graph-based approach to classification which is perhaps more related to ours is the work of Zhu et al. (2003) on binary classification with harmonic functions. Indeed,

the Gaussian field considered there is similar to the Potts model we obtain on the junction nodes. The approach of Zhu et al. (2003) is however just concerned by inference and not by learning, but their approach could be extended both to multi-class classification and to perform learning of the parameters.

8 EXPERIMENTS

We present in this section experiments on real data. Synthetic experiments on the core model of the CGMRF (without hidden layer) can be found in section 6 of the appendix.

In geographic information systems, data is often aggregated either on regular grid or on cells corresponding to abstract administrative boundaries, which do not necessarily reflect the structure of a city. A fairly natural type of representation for urban environment is based on graphs and in particular weighted graphs which can encode a distance information.

We consider a problem from geomatics in which this type of representation could be beneficial and which consists in predicting building use in urban and peri-urban environments from a few annotations and simple building shape characteristics that can be extracted easily from aerial images. More precisely, we consider the transductive learning problem of assigning city blocks to one category from $\{\textit{individual housing}, \textit{collective housing}, \textit{industrial/commercial area}\}$.

8.1 Building the city block continuous graph

A city can be divided into city blocks using its layout and road network as in Figure 8.1. Assuming that the blocks are given, we compute the Voronoi diagram of the block centroids and link together blocks with adjacent Voronoi cells. Edges are annotated with a proximity measure, in our case the distance between their respective closest buildings. This provides a continuous graph encapsulating the structure of the city. Each block is then annotated into one of three categories : individual residential, collective residential and industrial/commercial area. The blocks are annotated by hand using cadastral information, business registration codes, and resorting to Google street view images for ambiguous blocks (see Figure 8.1).

8.2 Data descriptors and learning setting

A block is then described by the weighted average of characteristics of the buildings it contains, each building counting with a weight proportional to its volume. We tested 10 different building descriptors, found that floor area and height were the most discriminative, and that adding more descriptors actually decreases the performance of all tested algorithms.

We use the example of Sevrans, a French city of 50 000 inhabitants north of Paris. We divided it into 461 blocks,

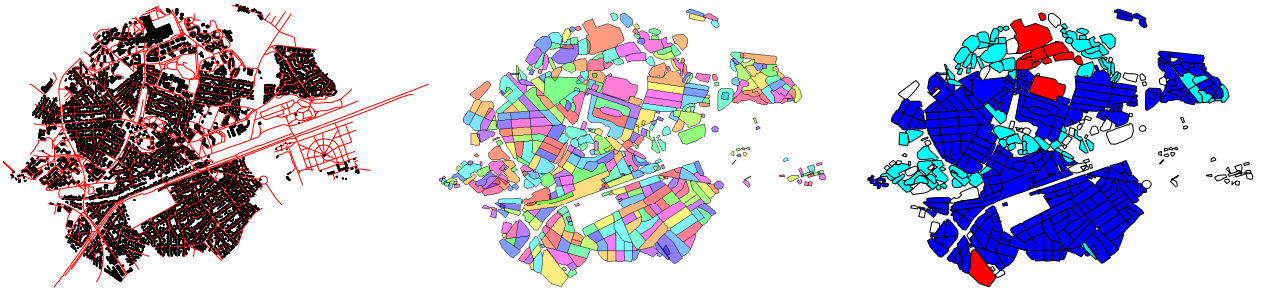


Figure 8.1: (left) Buildings and road network of Sevan. (middle) Division into city blocks. (right) City blocks with annotations. Blue: individual housing, cyan: collective housing, red: industrial/commercial area. (Best seen in color.)

400 of which can clearly be assigned one of three labels mentioned above and the rest being of insignificant size, ambiguous, or corresponding to other categories such as schools or hospitals.

We consider the transductive learning problem of predicting all block labels from a subset of labelled blocks. In our experiments, 7% of annotated labels, corresponding to 28 blocks, are used for training and the remaining are used for testing.

8.3 Competing algorithms

As baselines we consider two algorithms that do not take into account spatial information: a generative Gaussian mixture model and a logistic regression trained each using the 7% revealed labels. We also consider classical hidden MRFs, which cannot take into account the distance, and whose graph is either the same as for the CGMRF or a pruned graph in which all edges longer than a threshold (corresponding to the average city block radius) have been removed. The different graphs are illustrated on Figure 8.2. Note that the Gaussian mixture model does not take the graph structure into account, and can be interpreted as an edgeless MRF

In all Markov models, we use Gaussian emissions to model the distribution of the building descriptors given the block label, which can conveniently be optimized in closed form. To train the CGMRF and MRF models we learn the parameter θ with the maximum likelihood principle following the approach presented in section 6.5.

8.4 Results analysis

For each model, we construct a precision-coverage curve, obtained by sorting the probabilistic predictions by increasing values of their entropies, and reported on Figure 8.3. The confidence bands represented corresponds to one standard error for the estimation of the mean precision.

We can see that enriching the simple Gaussian mixture model by adding a graph structure significantly improves the overall performance. Building a MRF using all the edges from the Voronoi proximity or only retaining a fraction of the shorter edges yields similar results, on par with logistic regression. Building a CMRF using the edges annotated with a distance measure leads to a performance which is significantly above all others based on estimated standard errors.

When making prediction for all unlabeled points from the 7% of revealed annotations, the different algorithms yield the following average precisions (over the 300 resamplings): for the Gaussian mixture model 88.0%, for logistic regression 92.5%, the full MRF 92.4%, the pruned MRF 91.6% and our CGMRF 94.0%. Both pruned MRF and full MRF outperform the simple Gaussian mixture model, but not logistic regression, even though their precision at intermediate coverage is higher. The misclassification error of the CGMRF is 20% smaller than that of logistic regression, 21.5% smaller than for the best MRF model, and 50.2% smaller than for the Gaussian mixture. The gain in precision is not only obtained in average since the misclassification error in the CMRF was lower than MRF and logistic regression in respectively 193 and 293 out of 300 experiments. Wilcoxon signed rank tests assigns respectively p-values of $7 \cdot 10^{-26}$ and $3 \cdot 10^{-24}$ to the common median hypothesis.

In this experiment, with 461 nodes and 2718 edges the inference takes less than 0.1s on a CPU at 3.3GHz. Learning requires usually around 50 calls to the inference step for the MRF (5s total), while it is closer to 100 for the CGMRF (10s total).

8.5 Transfer learning on another city

We now consider the problem of predicting block labels on a new unannotated city using partial annotation from a given city. More precisely, we train our model with 15% of revealed labels from Sevan, and consider several

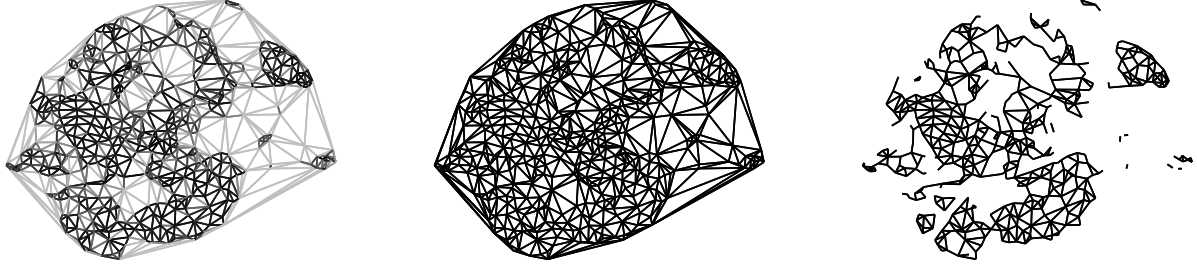


Figure 8.2: (left) continuous graph used to train the HCGMRF, the darker the edge the shorter the annotated distance, (middle) graph used for the HMRF including all edges or (right) with only edges shorter than a threshold.

schemes to make predictions on the neighboring urban area formed by Pierrefitte-sur-Seine together with Stains, for a total of 63000 inhabitants and 583 blocks, for which both graph and features are available but no labels are revealed. We consider logistic regression and the Gaussian mixture model trained from the annotated blocks from Sevrans as baselines, and test for each of the CGMRF and MRF the models learnt as follows:

- θ and κ are learnt on data from Sevrans
- idem followed by a single EM-step on κ alone (E-M2) on the graph of Pierrefitte+Stains
- idem followed by an EM-step on θ (E-M1) and then an EM-step on κ (E-M2).

We use the 359 labelled blocks (out of 583) of the Pierrefitte/Stains conglomeration as a testing set and construct the precision-coverage curves reported on Figure 8.4 (see the appendix for a figure comparing more approaches). We observe that the CGMRF setting is superior to its competitors, and that the relearning step improves the performance. The MRFs does not perform as well, which can be explained by the initial prediction being inferior, and relearning degrades its performance. The setting where only one E-M2 step is performed yields in both cases results comprised between the two other settings.

9 CONCLUSION

In this paper, we constructed a Potts model over a continuous graph and showed how to compute the likelihood of several of its variants as well as the corresponding gradients, for the purpose of learning.

Our experiments on a problem from geomatics show that this model outperforms regular MRFs, and compares favorably with logistic regression which although discriminative does not leverage unlabelled data. Finally, we showed that the model can be used to perform transfer learning from a first partially labelled graph towards a new completely unlabelled graph.

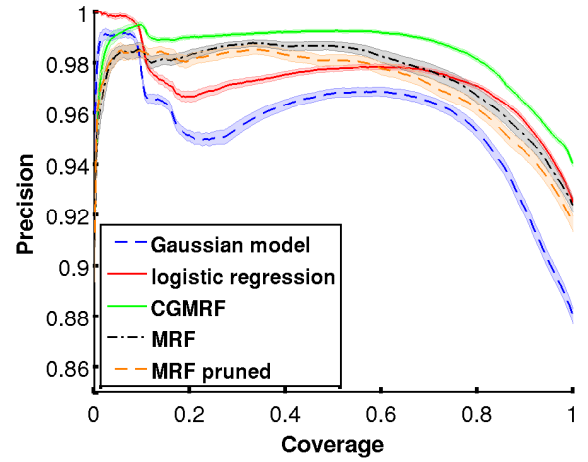


Figure 8.3: **Precision coverage curves on Sevrans.** Averaged precision coverage curves for the inference for 300 random resamplings of 7% of revealed labels on the city of Sevrans. (Best seen in color.)

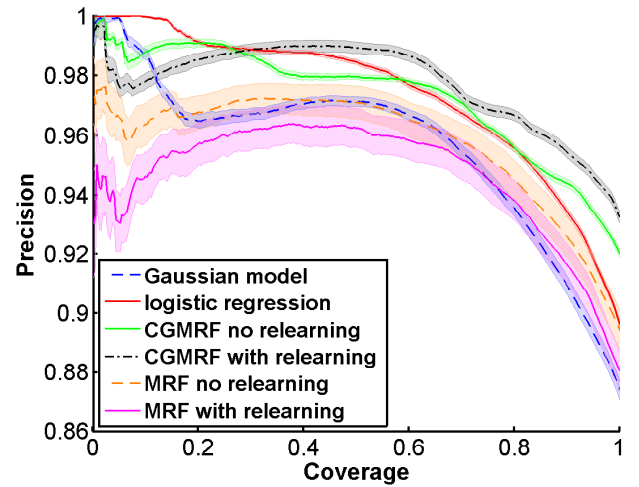


Figure 8.4: **Precision coverage curves for transfer learning.** Averaged precision coverage curves for the inference on the Pierrefitte/Stains conglomeration for 200 random resamplings of 15% of revealed labels on the city of Sevrans. (Best seen in color.)

References

- Bach, F. R. and Jordan, M. I. (2006). Learning spectral clustering, with application to speech separation. *The Journal of Machine Learning Research*, 7:1963–2001.
- Chiswell, I. (2001). *Introduction to Lambda Trees*. World Scientific Publishing Company.
- Chung, W. H. and Speyer, J. L. (1998). *Stochastic Processes, Estimation, and Control*. Society for Industrial and Applied Mathematics.
- Durrett, R. (2008). *Probability models for DNA sequence evolution*. Springer.
- Holmes, I. and Rubin, G. (2002). An expectation maximization algorithm for training hidden substitution models. *Journal of Molecular Biology*, 317(5):753–764.
- Nielsen, R. (2005). *Statistical methods in molecular evolution*. Springer.
- Nocedal, J. and Wright, S. (1999). *Numerical Optimization*. Springer.
- Nodelman, U., Shelton, C. R., and Koller, D. (2002). Continuous time Bayesian networks. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, pages 378–387. Morgan Kaufmann Publishers Inc.
- Norris, J. R. (1997). *Markov chains*. Cambridge University Press.
- Seber, G. A. (2008). *A matrix handbook for statisticians*, volume 15. Wiley.
- Von Bing, Y. and Speed, T. P. (2004). Modeling DNA base substitution in large genomic regions from two organisms. *Journal of Molecular Evolution*, 58(1):12–18.
- Wainwright, M. J. and Jordan, M. I. (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305.
- Yaple, H. A. and Abrams, D. M. (2013). A continuum generalization of the Ising model. *arXiv1306.3528*.
- Zhu, X., Ghahramani, Z., and Lafferty, J. (2003). Semi-supervised learning using Gaussian fields and harmonic functions. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 3, pages 912–919.
- Zhu, X. and Goldberg, A. B. (2009). Introduction to semi-supervised learning. *Synthesis lectures on Artificial Intelligence and Machine Learning*, 3(1):1–130.

Efficient Inference of Gaussian-Process-Modulated Renewal Processes with Application to Medical Event Data

Thomas A. Lasko

Computational Medicine Laboratory, Department of Biomedical Informatics
Vanderbilt University School of Medicine
Nashville, TN 37203

Abstract

The episodic, irregular and asynchronous nature of medical data render them difficult substrates for standard machine learning algorithms. We would like to abstract away this difficulty for the class of time-stamped categorical variables (or *events*) by modeling them as a renewal process and inferring a probability density over non-parametric longitudinal intensity functions that modulate the process. Several methods exist for inferring such a density over intensity functions, but either their constraints prevent their use with our potentially bursty event streams, or their time complexity renders their use intractable on our long-duration observations of high-resolution events, or both. In this paper we present a new efficient and flexible inference method that uses direct numeric integration and smooth interpolation over Gaussian processes. We demonstrate that our direct method is up to twice as accurate and two orders of magnitude more efficient than the best existing method (thinning). Importantly, our direct method can infer intensity functions over the full range of bursty to memoryless to regular events, which thinning and many other methods cannot do. Finally, we apply the method to clinical event data and demonstrate a simple example application facilitated by the abstraction.

1 INTRODUCTION

One of the hurdles for identifying clinically meaningful patterns in medical data is the fact that much of that data is sparsely, irregularly, and asynchronously observed, rendering it a poor substrate for many pattern recognition algorithms.

A large class of this problematic data in medical records is time-stamped categorical data such as billing codes.

For example, an ICD-9 billing code with categorical label 714.0 (Rheumatoid Arthritis) gets attached to a patient record every time the patient makes contact with the healthcare system for a problem or activity related to her arthritis. The activity could be an outpatient doctor visit, a laboratory test, a physical therapy visit, a discharge from an inpatient stay, or any other billable event. These events occur at times that are generally asynchronous with events for other conditions.

We would like to learn things from the patterns of these clinical contact events both within and between diseases, but their often sparse and irregular nature makes it difficult to apply standard learning algorithms to them. To abstract away this problem, we consider the data as streams of events, one stream per code or other categorical label. We model each stream as a modulated renewal process and use the process's modulation function as the abstract representation of the label's activity. The modulation function provides continuous longitudinal information about the intensity of the patient's contact with the healthcare system for a particular problem at any point in time. Our goal is to infer these functions, the renewal-process parameters, and the appropriate uncertainties from the raw event data.

We have previously demonstrated the practical utility of using a continuous function density to couple standard learning algorithms to sparse and irregularly observed continuous variables (Lasko et al., 2013). Unfortunately, the method of inferring such densities for continuous variables is not applicable to categorical variables. This paper presents a method that achieves the inference for categorical variables.

Our method models the log intensity functions nonparametrically as Gaussian processes, and uses Markov Chain Monte Carlo (MCMC) to infer a posterior distribution over intensity functions and model parameters given the events (Section 2).

There are several existing approaches to making this inference (Section 3), but all of the approaches we found have either flexibility or scalability problems with our clinical

data. For example, clinical event streams can be bursty, and some existing methods are unable to adapt to or adequately represent bursty event patterns.

In this paper we demonstrate using synthetic data that our approach is up to twice as accurate, up to two orders of magnitude more efficient, and more flexible than the best existing method (Section 4.1). We also demonstrate our method using synthetic data that mimics our clinical data, under conditions that no existing method that we know of is able to satisfactorily operate (Section 4.1). Finally, we use our method to infer continuous abstractions over real clinical data (Section 4.2), and as a simple application example we infer latent compound diseases from a complex patient record that closely correspond to its documented clinical problems.

2 MODULATED RENEWAL PROCESS EVENT MODEL

A renewal process models random events by assuming that the interevent intervals are independent and identically distributed (iid). A modulated renewal process model drops the iid assumption and adds a longitudinal intensity function that modulates the expected event rate with respect to time.

We consider a set of event times $T = \{t_0, t_1, \dots, t_n\}$ to form an event stream that can be modeled by a modulated renewal process. For this work we choose a modulated gamma process (Berman, 1981), which models the times T as

$$\mathbf{P}(T; a, \lambda(t)) = \frac{1}{\Gamma(a)^n} \prod_{i=1}^n \lambda(t_i) (\Lambda(t_i) - \Lambda(t_{i-1}))^{a-1} e^{-\Lambda(t_n)}, \quad (1)$$

where $\Gamma(\cdot)$ is the gamma function, $a > 0$ is the shape parameter, $\lambda(t) > 0$ is the modulating intensity function, and $\Lambda(t) = \int_0^t \lambda(u) du$.

Equation (1) is a generalization of the *homogeneous* gamma process $\gamma(a, b)$, which models the interevent intervals $x_i = t_i - t_{i-1}$, $i = 1 \dots n$ as positive iid random variables:

$$\gamma(x|a, b) = \mathbf{P}(x; a, b) = \frac{1}{\Gamma(a)b^a} x^{a-1} e^{-x/b}, \quad (2)$$

where b takes the place of a now-constant $1/\lambda(t)$, and can be thought of as the time scale of event arrivals.

The intuition behind (1) is that the function $\Lambda(t)$ warps the event times t_i into a new space where their interevent intervals become draws from the homogeneous gamma process of (2). That is, the warped intervals $w_i = \Lambda(t_i) - \Lambda(t_{i-1})$ are modeled by $w_i \sim \gamma(a, b)$.

For our purposes, a gamma process is better than the simpler and more common Poisson process because a gamma process allows us to model the relationship between neighboring events, instead of assuming them to be independent or memoryless. Specifically, parameterizing $a < 1$ models a bursty process, $a > 1$ models a more regular or refractory process, and $a = 1$ produces the memoryless Poisson process. Clinical event streams can behave anywhere from highly bursty to highly regular.

We model the log intensity function $\log \lambda(t) = f(t) \sim \mathcal{GP}(0, C)$ as a draw from a Gaussian process prior with zero mean and the squared exponential covariance function

$$C(t_i, t_j) = \sigma e^{-\left(\frac{t_i - t_j}{l}\right)^2}, \quad (3)$$

where σ sets the magnitude scale and l sets the time scale of the Gaussian process. We choose the squared exponential because of its smoothness guarantees that are relied upon by our inference algorithm, but other covariance functions could be used.

In our application the observation period generally starts at $t_{\min} < t_0$, and ends at $t_{\max} > t_n$, and no events occur at these endpoints. Consequently, we must add terms to (1) to account for these partially observed intervals. For efficiency in inference, we estimate the probabilities of these intervals by assuming that w_0 and w_{n+1} are drawn from a homogeneous $\gamma(1, 1)$ process in the warped space. The probability of the leading interval $w_0 = \Lambda(t_0) - \Lambda(t_{\min})$ is then approximated by $\mathbf{P}(w \geq w_0) = \int_{w_0}^{\infty} e^{-w} dw = e^{-w_0}$, which is equivalent to $w_0 \sim \gamma(1, 1)$. The trailing interval is treated similarly.

Our full generative model is as follows:

1. $l \sim \text{Exponential}(\alpha)$
 $\log \sigma \sim \text{Uniform}(\log \sigma_{\min}, \log \sigma_{\max})$
 $\log a \sim \text{Uniform}(\log a_{\min}, \log a_{\max})$
 $b = 1$
2. $f(t) \sim \mathcal{GP}(0, C)$ using (3)
3. $\lambda(t) = e^{f(t)}$
4. $\Lambda(t) = \int_0^t \lambda(u) du$
5. $w_0 \sim \gamma(1, 1)$; $w_{i>0} \sim \gamma(a, b)$
6. $t_i = t_{\min} + \Lambda^{-1}(\sum_{j=0}^i w_j)$

Step 1 places a prior on l that prefers smaller values, and uninformative priors on a and σ . We set $b = 1$ to avoid an identifiability problem. (Rao and Teh (2011) set $b = 1/a$ to avoid this problem. While that setting has some desirable properties, we've found that setting $b = 1$ avoids more degenerate solutions at inference time.)

2.1 INFERENCE

Given a set of event times T , we use MCMC to simultaneously infer posterior distributions over the intensity func-

tion $\lambda(T)$ and the parameters a , σ , and l (Algorithm 1). For simplicity we denote $g(T) = \{g(t) : t \in T\}$ for any function g that operates on event times. On each round we first use slice sampling with surrogate data (Murray and Adams, 2010, code publicly available) to compute new draws of $f(t)$, σ , and l using (1) as the likelihood function (with additional factors for the incomplete interval at each end). We then sample the gamma shape parameter a using Metropolis-Hastings updates.

One challenge of this direct inference is that it requires integrating $\Lambda(t) = \int_0^t \lambda(u) du$, which is difficult because $\lambda(t)$ does not have an explicit expression. Under certain conditions, the integral of a Gaussian process has a closed form (Rasmussen and Ghahramani, 2003), but we know of no closed form for the integral of a log Gaussian process. Instead, we compute the integral numerically (the trapezoidal rule works fine), relying on the smoothness guarantees provided by the covariance function (3) to provide high accuracy.

The efficiency bottleneck of the update is the $O(m^3)$ complexity of updating the Gaussian process f at m locations, due to a matrix inversion. Naively, we would compute f at all n of the observed t_i , with additional points as needed for accuracy of the integral. To improve efficiency, we do not directly update f at the t_i , but instead at k uniformly spaced points $\hat{T} = \{\hat{t}_j = t_{\min} + jd\}$, where $d = \frac{t_{\max} - t_{\min}}{k-1}$. We then interpolate the values $f(T)$ from the values of $f(\hat{T})$ as needed. We set the number of points k by the accuracy required for the integral. This is driven by our estimate of the smallest likely Gaussian process time scale l_{\min} , at which point we truncate the prior on l to guarantee $d \ll l_{\min} \leq l$. The efficiency of the resulting update is $O(k^3) + O(n)$, with k depending only on the ratio $l_{\min}/(t_{\max} - t_{\min})$.

It helps that the factor driving k is the time scale of changes in the intensity function $\lambda(t)$ instead of the time scale of interevent intervals, which is usually much smaller. In practice, we’ve found $k = 200$ to work well for nearly all of our medical data examples, regardless of the observation time span, resulting in an update that is linear in the number of observed points.

Additionally, the regular spacing in \hat{T} means that its covariance matrix generated by (3) is a symmetric positive definite Toeplitz matrix, which can be inverted or solved in a compact representation as fast as $O(k \log^2 k)$ (Martinson et al., 2005). We did not include this extra efficiency in our implementation, however.

3 RELATED WORK

There is a growing literature on finding patterns among clinical variables such as laboratory tests that have both a timestamp and a numeric value (Lasko et al., 2013), but we are not aware of any existing work exploring unsuper-

Algorithm 1 Intensity Function and Parameter Update

Input: Event times T , regular grid \hat{T} , current function f and parameters σ , l , and a

Output: Updated f , λ , Λ , σ , l , and a with likelihood p

- 1: Update $f(\hat{T})$, σ , l , using slice sampling
 - 2: Compute $f(T)$ by smooth interpolation of $f(\hat{T})$
 - 3: $\lambda(T \cup \hat{T}) \leftarrow e^{f(T \cup \hat{T})}$
 - 4: Compute $\Lambda(T)$ from $\lambda(\hat{T})$ numerically
 - 5: Compute $p = P(T; a, \lambda(T))$ using (1)
 - 6: Update a and p with Metropolis-Hastings and (1)
-

vised, data-driven abstractions of categorical clinical event streams that we address here.

There is much prior work on methods similar to ours that infer intensity functions for modulated renewal processes. The main distinction between these methods lies in the way they handle the form and integration of the intensity function $\lambda(t)$. Approaches include using kernel density estimation (Ramlau-Hansen, 1983), using parametric intensity functions (Lewis, 1972), using discretized bins within which the intensity is considered constant (Moller et al., 1998, Cunningham et al., 2008), or using a form of rejection sampling called *thinning* (Adams et al., 2009, Rao and Teh, 2011) that avoids the integration altogether.

The binned time approach is straightforward, but there is inherent information loss in the piecewise-constant intensity approximation that it must adopt. Moreover, when a Gaussian process is used to represent this intensity function, the computational complexity of inference is cubic with the number of bins in the interval of observation. For our data, with events at 1-day or finer time resolution over up to a 15 year observation period, this method is prohibitively inefficient. A variant of the binned-time approach that uses variable-sized bins (Gunawardana et al., 2011, Parikh et al., 2012) has been applied to medical data (Weiss and Page, 2013). This variant is very efficient, but is restricted to a Poisson process (fixed $a = 1$), and the inferred intensity functions are neither intended to nor particularly well suited to form an accurate abstraction over the raw events.

Thinning is a clever method, but it is limited by the requirement of a bounded hazard function, which prevents it from being used with bursty gamma processes. (Bursty gamma processes have a hazard function that is unbounded at zero). One thinning method has also adopted the use of Gaussian processes to modulate gamma processes (Rao and Teh, 2011). But in addition to not working with bursty events, it also rather inefficient; its time complexity is cubic in the number of events that would occur if the maximum event intensity were constant over the entire observation time span. For event streams with a small dynamic range of intensities, this is not a big issue, but our medical data

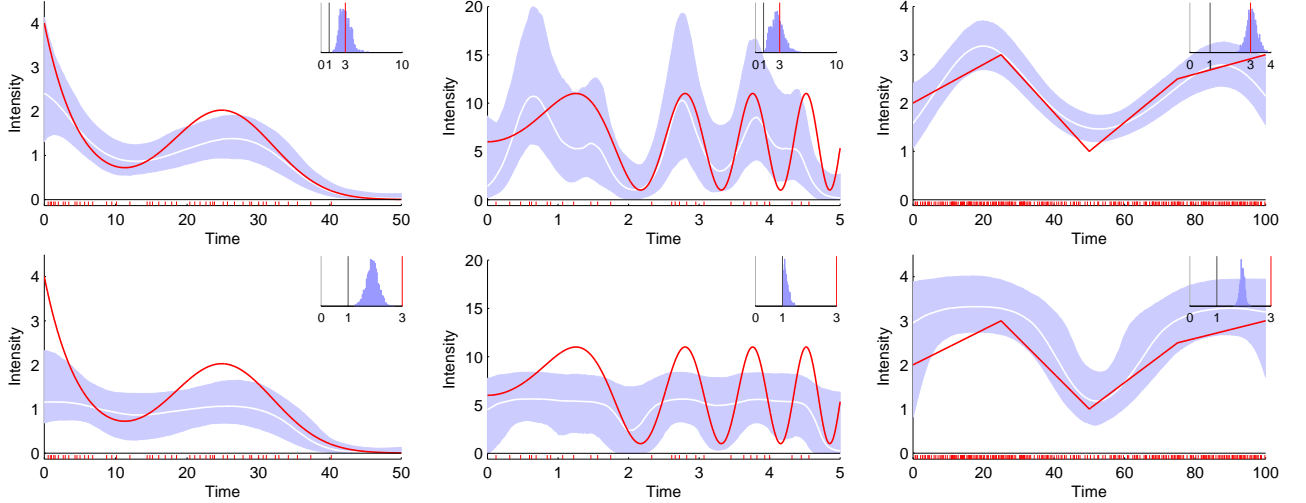


Figure 1: Our direct method (top) is more accurate than thinning (bottom) on parametric intensity functions λ_1 to λ_3 (left to right). Red line: true normalized intensity function $\lambda(t)/a$; White line: mean inferred normalized intensity function; Blue region: 95% confidence interval. Inset: inferred distribution of the gamma shape parameter a , with the true value marked in red. Grey bar at $a = 1$ for reference.

sequences can have a dynamic range of several orders of magnitude.

Our method therefore has efficiency and flexibility advantages over existing methods, and we will demonstrate in the experiments that it also has accuracy advantages.

4 EXPERIMENTS

In these experiments, we will refer to our inference method as the *direct method* because it uses direct numerical integration. A full implementation of our method and code to reproduce the results on the synthetic data is available at <https://github.com/ComputationalMedicineLab/egpmrp>.

We tested the ability of the direct method and the thinning method to recover known intensity functions and shape parameters from synthetic data. We then used the direct method to extract latent intensity functions from streams of clinical events, and we inferred latent compound conditions from the intensity functions for a complex patient record that accurately correspond to the dominant diseases documented in the record.

4.1 SYNTHETIC DATA

Our first experiments were with the three parametric intensity functions below, carefully following Adams et al. (2009) and Rao and Teh (2011). We generated all data using the warping model described in Section 2, with shape parameter $a = 3$.

1. $\lambda_1(t)/a = 2e^{-t/15} + e^{-((t-25)/10)^2}$ over the interval

$[0, 50]$, 48 events.

2. $\lambda_2(t)/a = 5 \sin(t^2) + 6$ on $[0, 5]$, 29 events.

3. $\lambda_3(t)/a$ is the piecewise linear curve shown in Figure 1, on the interval $[0, 100]$, 230 events.

We express these as normalized intensities $\lambda(t)/a$, which have units of “expected number of events per unit time”, because they are more interpretable than the raw intensities and they are comparable to the previous work done using Poisson processes, where $a = 1$.

We compared the direct method to thinning on these datasets, using the MATLAB implementation for thinning that was used by Rao and Teh (2011). Adams et al. (2009) compared thinning to the kernel smoothing and binned time methods (all assuming a Poisson process), and Rao and Teh (2011) compared thinning to binned time, assuming a gamma process with constrained $a > 1$. Both found thinning to be at least as accurate as the other methods in most tests.

We computed the RMS error of the true vs. the median normalized inferred intensity, the log probability of the data given the model, and the inference run time under 1000 burn-in and 5000 inference MCMC iterations.

On these datasets the direct method was more accurate than thinning for the recovery of both the intensity function and the shape parameter, and more efficient by up to two orders of magnitude (Figure 1 and Table 1). The results for thinning are consistent with those previously reported (Rao and Teh, 2011), with the exception that the shape parameter inference was more accurate in the earlier results.

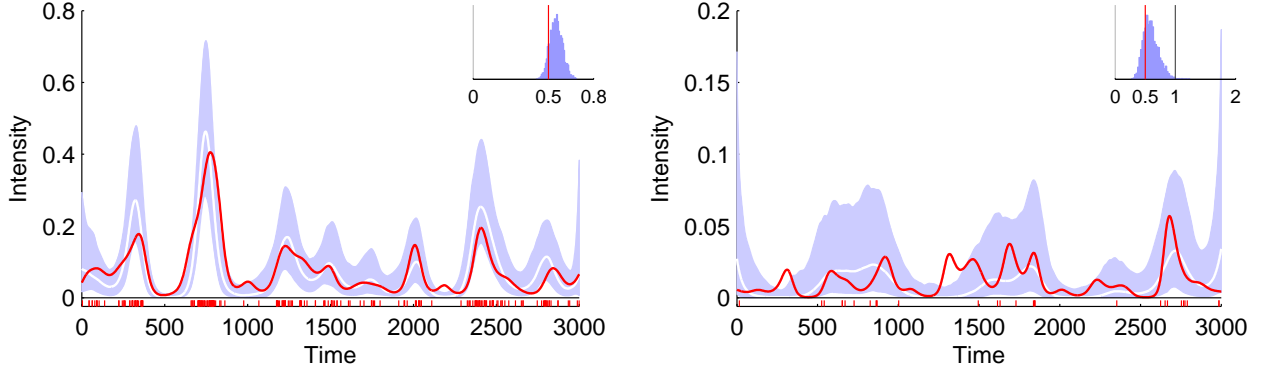


Figure 2: Accurate recovery of intensity function and parameters under conditions that would be prohibitive for any other method of which we are aware. Left panel presents results for high intensities and many events, right panel for low intensities and few events. While there is insufficient evidence in the right panel to recover the true intensity, the inferred intensity is reasonable given the evidence, and the inferred confidence intervals are accurate in that the true intensity is about 95% contained within them. Legend as in Figure 1.

The confidence intervals from the direct method are subjectively more accurate than from the thinning method. (That is, the 95% confidence intervals from the direct method contain the true function for about 95% of its length in each case). This is particularly important in the case of small numbers of events that may not carry sufficient information for any method to resolve a highly varying function.

As might be expected, we found the results for $\lambda_2(t)$ to be sensitive to the prior distribution on l , given the small amount of evidence available for the inference. Following Adams et al. (2009) and Rao and Teh (2011), we used a log-normal prior with a mode near $l = 0.2$, tuned slightly for each method to achieve the best results. We also allowed thinning to use a log-normal prior with appropriate modes for $\lambda_1(t)$ and $\lambda_3(t)$, to follow precedent in the previous work, although it may have conferred a small advantage to thinning. We used the weaker exponential prior on those datasets for the direct method.

Our next experiments were on synthetic data generated to resemble our medical data. We tested several configurations over wide ranges of parameters, including some that were not amenable to any known existing approach (such as the combination of $a < 1$, high dynamic range of in-

tensity, and high ratio of observation period to event resolution, Figure 2). The inferred intensities and gamma parameters were consistently accurate. Estimates of the confidence intervals were also accurate, including in cases with low intensities and few events (Figure 2, right panel).

4.2 CLINICAL DATA

We applied the direct method to sequences of billing codes representing clinical events. After obtaining IRB approval, we extracted all ICD-9 codes from five patient records with the greatest number of such codes in the deidentified mirror of our institution’s Electronic Medical Record. We arranged the codes from each patient record as streams of events grouped at the top (or *chapter*) level of the ICD-9 disease hierarchy (which collects broadly related conditions), as well as at the level of the individual disease.

For the streams of grouped events, we included an event if its associated ICD-9 code fell within the range of the given top-level division. For example, any ICD-9 event with a code in the range [390 – 459.81] was considered a *Cardiovascular* event. While intensity functions are only strictly additive for Poisson processes, we still find the curves of grouped events to be informative.

We inferred intensity functions for each of these event streams (for example, Figure 3). Each curve was generated using 2000 burn-in and 2000 inference iterations in about three minutes using unoptimized MATLAB code on a single desktop CPU. The results have good clinical face validity.

There is much underlying structure in these events that can now be investigated with standard learning methods applied to the inferred intensity functions. As a simple example, singular value decomposition (Strang, 2003) can infer the latent compound conditions (which we might as well

Table 1: Performance on Synthetic Data. RMS: root-mean-squared error; LP: log probability of data given the model; RT: run time in seconds. Best results for each measure are bolded.

	Direct			Thinning		
	RMS	LP	RT	RMS	LP	RT
λ_1	0.37	+12.1	453	0.66	−62.7	4816
λ_2	3.1	−228	511	3.4	−333	1129
λ_3	0.25	+1.21	385	0.53	−82.2	41291

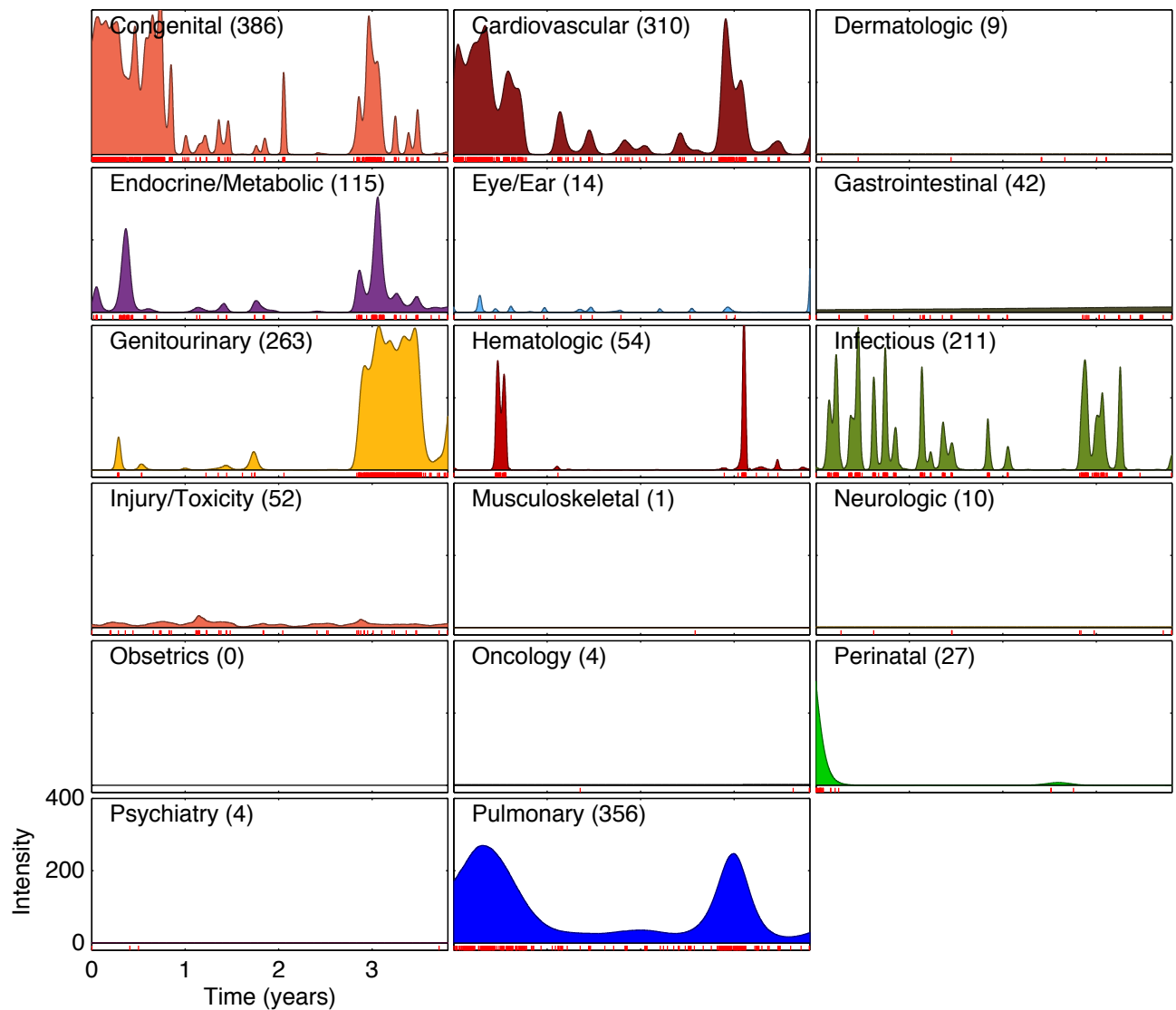


Figure 3: Inferred intensities for all top-level ICD-9 disease divisions of a very complicated patient's record. Such a display may be clinically useful for getting a quick, broad understanding of a patient's medical history, including quickly grasping which conditions have *not* been diagnosed or treated. Numbers in parentheses: total number of events in each division. For clarity, confidence intervals are not shown.

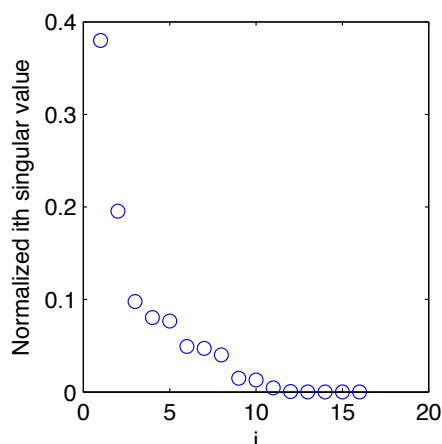


Figure 4: A small number of latent compound conditions (or *eigendiseases*) produce most of the activity captured by the patient record in Figure 3.

call *eigendiseases*) underlying the recorded clinical activity, taking into account the continuously changing, longitudinal time course of that activity. The singular values for the curves in Figure 3 reveal that about 40% of the patient’s activity relates to a single eigendisease, and about 70% is distributed among the top three (Figure 4). The inferred eigendiseases closely correspond to the dominant clinical problems described in the record (Figure 5).

5 DISCUSSION

We have made two contributions with this paper. First, we presented a direct numeric method to infer a distribution of continuous intensity functions from a set of episodic, irregular, and discrete events. This direct method has increased efficiency, flexibility, and accuracy compared to the best prior method. Second, we presented results using the direct method to infer a continuous function density as an abstraction over episodic clinical events, for the purposes of transforming the raw event data into a form more amenable to standard machine learning algorithms.

The clinical interpretation of these intensity functions is that increased intensity represents increased frequency of contact with the healthcare system, which usually means increased *instability* of that condition. In some cases, it may also mean increased *severity* of the condition, but not always. If a condition acutely increases in severity, this represents an instability and will probably generate a contact event. On the other hand, if a condition is severe but stably so, it may not necessarily require high-frequency medical contact.

The methods described here to represent categorically labeled events in time as continuous curves augment our

previously reported methods to construct similar curves from observations with both a time and a continuous value (Lasko et al., 2013). These two data types represent the majority of the information in a patient record (if we consider words and concepts in narrative text to be categorical variables), and opens up many possibilities for finding meaningful patterns in large medical datasets.

The practical motivation for this work is that once we have the continuous function densities, we can use them as inputs to a learning problem in the time domain (such as identifying trajectories that may be characteristic of a particular disease), or by aligning many such curves in time and looking for useful patterns in their cross-sections (which to our knowledge has not yet been reported). We presented a simple demonstration of inferring cross-sectional latent factors from the intensity curves of a single record. Discovering similar latent factors underlying a large population is a focus of future work.

We discovered incidentally that a presentation such as Figure 3 appears to be a promising representation for efficiently summarizing a complicated patient’s medical history and communicating that broad summary to a clinician. The presentation could allow drilling-down to the intensity plots of the specific component conditions and then to the raw source data. (The usual method of manually paging through the often massive chart of a patient to get this information can be a tedious and frustrating process.)

One could also imagine presenting the curves not of the raw ICD-9 codes, but of the inferred latent factors underlying them, and drilling down into the rich combinations of test results, medications, narrative text, and discrete billing events that comprise those latent factors.

We believe that these methods will facilitate *Computational Phenotype Discovery* (Lasko et al., 2013), or the data-driven search for population-scale clinical patterns in existing electronic medical records that may illuminate previously unknown disease variants, unanticipated medication effects, or emerging syndromes and infectious disease outbreaks.

Acknowledgements

This work was funded by grants from the Edward Mallinckrodt, Jr. Foundation and the National Institutes of Health 1R21LM011664-01. Clinical data was provided by the Vanderbilt Synthetic Derivative, which is supported by institutional funding and by the Vanderbilt CTSA grant ULTR000445. Thanks to Vinayak Rao for providing the thinning code.

References

R. P. Adams, I. Murray, and D. J. C. MacKay. Tractable nonparametric Bayesian inference in Poisson processes

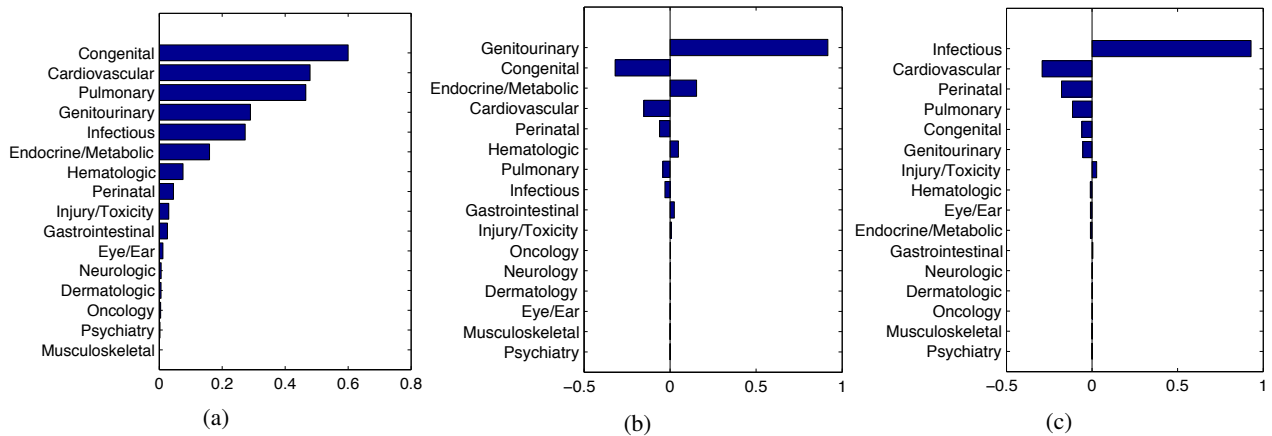


Figure 5: The top three eigendiseases inferred from the patient record in Figure 3. These align well with the actual clinical problems described in the record, which are a) severe congenital malformation of the heart and lungs with downstream effects on multiple organ systems, b) non-congenital kidney failure (a genitourinary condition with metabolic consequences), and c) multiple recurring infections from several sources.

- with Gaussian process intensities. In *Proceedings of the 26th International Conference on Machine Learning*, 2009.
- M. Berman. Inhomogeneous and modulated gamma processes. *Biometrika*, 68(1):143 – 152, 1981. ISSN 00063444.
- J. P. Cunningham, B. M. Yu, and K. V. Shenoy. Inferring neural firing rates from spike trains using gaussian processes. In *Advances in Neural Information Processing Systems*, 2008.
- A. Gunawardana, C. Meek, and P. Xu. A model for temporal dependencies in event streams. In *Advances in Neural Information Processing Systems*, 2011.
- T. A. Lasko, J. C. Denny, and M. A. Levy. Computational phenotype discovery using unsupervised feature learning over noisy, sparse, and irregular clinical data. *PLoS One*, 8(6):e66341, 2013.
- P. A. W. Lewis. Recent results in the statistical analysis of univariate point processes. In P. A. W. Lewis, editor, *Stochastic Point Processes; Statistical Analysis, Theory, and Applications*. Wiley Interscience, 1972.
- P. Martinsson, V. Rokhlin, and M. Tytgert. A fast algorithm for the inversion of general toeplitz matrices. *Computers & Mathematics with Applications*, 50(5–6):741–752, 2005.
- J. Moller, A. R. Syversveen, and R. P. Waagepetersen. Log gaussian cox processes. *Scand J Stat*, 25(3):pp. 451–482, 1998.
- I. Murray and R. P. Adams. Slice sampling covariance hyperparameters of latent Gaussian models. In J. Lafferty, C. K. I. Williams, R. Zemel, J. Shawe-Taylor, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, 2010.
- A. P. Parikh, A. Gunawardana, and C. Meek. Conjoint modeling of temporal dependencies in event streams. In *UAI Bayesian Modelling Applications Workshop*, 2012.
- H. Ramlau-Hansen. Smoothing counting process intensities by means of kernel functions. *The Annals of Statistics*, 11(2):453 – 466, 1983. ISSN 00905364.
- V. Rao and Y. W. Teh. Gaussian process modulated renewal processes. In *Advances In Neural Information Processing Systems*, 2011.
- C. E. Rasmussen and Z. Ghahramani. Bayesian monte carlo. In *Advances in Neural Information Processing Systems*, 2003.
- G. Strang. Singular value decomposition (SVD). In *Introduction to Linear Algebra*, chapter 6.7, pages 352–362. Wellesley-Cambridge, Wellesley, MA, third edition, 2003.
- J. C. Weiss and D. Page. Forest-based point process for event prediction from electronic health records. In H. Blockeel, K. Kersting, S. Nijssen, and F. elezn, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 8190 of *Lecture Notes in Computer Science*, pages 547–562. Springer Berlin Heidelberg, 2013.

Optimal Resource Allocation with Semi-Bandit Feedback

Tor Lattimore

Dept. of Computing Science
University of Alberta, Canada

Koby Crammer

Dept. of Electrical Engineering
The Technion, Israel

Csaba Szepesvári*

Microsoft Research
Redmond, USA

Abstract

We study a sequential resource allocation problem involving a fixed number of recurring jobs. At each time-step the manager should distribute available resources among the jobs in order to maximise the expected number of completed jobs. Allocating more resources to a given job increases the probability that it completes, but with a cut-off. Specifically, we assume a linear model where the probability increases linearly until it equals one, after which allocating additional resources is wasteful. We assume the difficulty of each job is unknown and present the first algorithm for this problem and prove upper and lower bounds on its regret. Despite its apparent simplicity, the problem has a rich structure: we show that an appropriate optimistic algorithm can improve its learning speed dramatically beyond the results one normally expects for similar problems as the problem becomes resource-laden.

1 INTRODUCTION

Assume that there are K jobs and at each time-step t a learner must distribute the available resources with $M_{k,t} \geq 0$ going to job k , subject to a budget constraint,

$$\sum_{k=1}^K M_{k,t} \leq 1.$$

The probability that the k th job completes in time-step t is $\min\{1, M_{k,t}/\nu_k\}$, where the unknown cut-off parameter $\nu_k \in (0, \infty]$ determines the difficulty of job k . After every time-step the resources are replenished and all jobs are restarted regardless of whether or not they completed successfully in the previous time-step. The goal of the learner

is to maximise the expected number of jobs that successfully complete up to some known time horizon n .

Despite the simple model, the problem is surprisingly rich. Given its information structure, the problem belongs to the class of stochastic partial monitoring problems, which was first studied by [Agrawal et al. \[1989\]](#)¹, where in each time step the learner receives noisy information about a hidden “parameter” while trying to maximise the sum of rewards and both the information received and the rewards depend in a known fashion on the actions and the hidden parameter. While partial monitoring by now is relatively well understood, either in the stochastic or the adversarial framework when the action set is finite [[Bartók et al., 2011](#), [Foster and Rakhlin, 2012](#), [Bartók, 2013](#)], the case of continuous action sets has received only limited attention [[Broder and Rusmevichientong, 2012](#), and references therein]. To illustrate the difficulty of the problem, notice that over-assigning resources to a given job means that the job completes with certainty and provides little information about the job’s difficulty. On the other hand, if resources are under-assigned, then the information received allows one to learn about the payoff associated with all possible arms, which is reminiscent of bandit problems where the arms have “correlated payoffs” (e.g., [Filippi et al. 2010](#), [Russo and Roy 2013](#) and the references therein). Finally, allocating less resources yields high-variance estimates.

Our motivation to study this particular framework comes from the problem of cache allocation. In particular, data collected offline from existing and experimental allocation strategies showed a relatively good fit to the above parametric model. In this problem each job is a computer process, which is successful in a given time-step if there were no cache misses (cache misses are very expensive). Besides this specific resource allocation problem, we also envision other applications, such as load balancing in networked environments, or any other computing applications where some precious resource (bandwidth, radio spectrum, CPU, etc.) is to be subdivided amongst competing processes. In fact, we anticipate numerous extensions and adaptations for

*On sabbatical leave from the Department of Computing Science, University of Alberta, Canada

¹The name was invented later by (perhaps) [[Rustichini, 1999](#)].

specific applications, such as in the case of bandits (see, [Bubeck and Cesa-Bianchi \[2012\]](#) for an overview of this rich literature). Finally, let us point out that although our problem is superficially similar to the so-called budgeted bandit problems (or, budget limited bandit problems), there are some major differences: in budgeted bandits, the information structure is still that of bandit problems and the resources are not replenished. Either learning stops when the budget is exhausted (e.g., [Tran-Thanh et al. 2012](#), [Ding et al. 2013](#), [Badanidiyuru et al. 2013](#))², or performance is measured against the total resources consumed in an ongoing fashion (e.g., [Györfy et al. 2007](#)).

The main contribution besides the introduction of a new problem is a new optimistic algorithm for this problem that is shown to suffer poly-logarithmic regret with respect to optimal omniscient algorithm that knows the parameters $(\nu_k)_k$ in advance. The structure of the bound depends significantly on the problem dynamics, ranging from a (relatively) easy full-information-like setting, corresponding to a resource-laden regime, to a bandit-like setting, corresponding to the resource-scant setting. Again, to contrast this work to previous works, note that the results we obtain for the full-information-like setting are distinct from those possible in the finite action case, where the full-information setting allows one to learn with finite regret [[Agrawal et al., 1989](#)]. On the technical side, we believe that our study and use of weighted estimators in situations where some samples are more informative than others might be of independent interest, too.

Problems of allocating resources to jobs were studied in the community of architecture and operating systems. [Liu et al. \[2004\]](#) build static profile-based allocation of L2-cache banks to different processes using their current miss rate data. [Suh et al. \[2002\]](#) proposed a hit-rate optimisation using hardware counters which used a model-based estimation of hit-rate vs allocated cache. However, they all assume the model is fully known and no learning is required. [Bitirgen et al. \[2008\]](#) used ANNs to predict individual program performance as a function of resources. Finally, [Ipek et al. \[2008\]](#) used reinforcement learning to allocate DRAM to multi-processors.

2 PRELIMINARIES

In each time-step t the learner chooses $M_{k,t} \geq 0$ subject to the constraint, $\sum_{k=1}^K M_{k,t} \leq 1$. Then all jobs are executed and $X_{k,t} \in \{0, 1\}$ indicates the success or failure of job k in time-step t and is sampled from a Bernoulli distribution with parameter $\beta(M_{k,t}/\nu_k) := \min\{1, M_{k,t}/\nu_k\}$. The goal is to maximise the expected number of jobs that successfully complete, $\sum_{k=1}^K \beta(M_{k,t}/\nu_k)$. We define the gaps $\Delta_{j,k} = \nu_j^{-1} - \nu_k^{-1}$. We assume throughout for conve-

²Besides [Badanidiyuru et al. \[2013\]](#), all works consider finite action spaces and unstructured reward functions.

nience, and without loss of generality, that $\nu_1 < \nu_2 < \dots < \nu_K$. It can be shown that the optimal allocation distributes the resources to jobs in increasing order of difficulty.

$$M_k^* = \min \left\{ 1 - \sum_{i=1}^{k-1} M_i^*, \nu_k \right\}.$$

We let ℓ be the number of jobs that are fully allocated under the optimal policy: $\ell = \max\{i : M_i^* = \nu_i\}$. The overflow is denoted by $S^* = M_{\ell+1}^*$, which we assume to vanish if $\ell = K$. The expected reward (number of completed jobs) when following the optimal allocation is

$$\sum_{k=1}^K \frac{M_k^*}{\nu_k} = \ell + \frac{S^*}{\nu_{\ell+1}},$$

where we define $\nu_{K+1} = \infty$ in the case that $\ell = K$. The (expected n -step cumulative) regret of a given allocation algorithm is the difference between the expected number of jobs that complete under the optimal policy and those that complete given the algorithm,

$$\begin{aligned} R_n = \mathbb{E} \left[\sum_{t=1}^n r_t \right], \quad r_t &= \sum_{k=1}^K \beta(M_k^*/\nu_k) - \sum_{k=1}^K \beta(M_{k,t}/\nu_k) \\ &= \left(\ell + \frac{S^*}{\nu_{\ell+1}} \right) - \sum_{k=1}^K \beta(M_{k,t}/\nu_k). \end{aligned}$$

Some proofs are omitted due to space constraints, but may be found in the supplementary material [[Lattimore et al., 2014](#)].

3 OVERVIEW OF ALGORITHM

We take inspiration from the optimal policy for known ν_k , which is to fully allocate the jobs with the smallest ν_k (easiest jobs) and allocate the remainder/overflow to the next easiest job. At each time-step t we replace the unknown ν_k by a high-probability lower bound $\underline{\nu}_{k,t-1} \leq \nu_k$. This corresponds to the optimistic strategy, which assumes that each job is as easy as reasonably possible. The construction of a confidence interval about ν_k is surprisingly delicate. There are two main challenges. First, the function $\beta(M_{k,t}/\nu_k)$ is non-differentiable at $M_{k,t} = \nu_k$, and for $M_{k,t} \geq \nu_k$ the job will always complete and little information is gained. This is addressed by always using a lower estimate of ν_k in the algorithm. The second challenge is that $M_{k,t}$ will vary with time, so the samples $X_{k,t}$ are not identically distributed. This would normally be unproblematic, since martingale inequalities can be applied, but the specific structure of this problem means that a standard sample average estimator is a little weak in the sense that its estimation accuracy can be dramatically improved. In particular, we will propose an estimator that is able to take advantage of the fact that the

variance of $X_{k,t}$ decreases to zero as $M_{k,t}$ approaches ν_k from below.

As far as the estimates are concerned, rather than estimate the parameters ν_k , it turns out that learning the reciprocal ν_k^{-1} is both more approachable and ultimately more useful for proving regret bounds. Fix k and let $M_{k,1}, \dots, M_{k,t} \leq \nu_k$ be a sequence of allocations with $M_{k,s} \leq \nu_k$ and $X_{k,s} \sim \text{Bernoulli}(M_{k,s}/\nu_k)$. Then a natural (unbiased) estimator of ν_k^{-1} is given by

$$\frac{1}{\hat{\nu}_{k,t}} := \frac{1}{t} \sum_{s=1}^t \frac{X_{k,s}}{M_{k,s}}.$$

The estimator has some interesting properties. First, the random variable $X_{k,s}/M_{k,s} \in [0, 1/M_{k,s}]$ has a large range for small $M_{k,s}$, which makes it difficult to control the error $\hat{\nu}_{k,t}^{-1} - \nu_k^{-1}$ via the usual Azuma/Bernstein inequalities. Secondly, if $M_{k,s}$ is close to ν_k , then the range of $X_{k,s}/M_{k,s}$ is small, which makes estimation easier. Additionally, the variance is greatly decreased for $M_{k,s}$ close to ν_k . This suggests that samples for which $M_{k,s}$ is large are more useful than those where $M_{k,s}$ is small, which motivates the use of the weighted estimator,

$$\frac{1}{\hat{\nu}_{k,t}} := \frac{\sum_{s=1}^t w_s X_{k,s}}{\sum_{s=1}^t w_s M_{k,s}},$$

where w_s will be chosen in a data-dependent way, but with the important characteristic that w_s is large for $M_{k,s}$ close to ν_k . The pseudo-code of the main algorithm is shown on Algorithm Listing 1. It accepts as input the horizon n , the number of jobs, and a set $\{\underline{\nu}_{k,0}\}_{k=1}^K$ for which $0 < \underline{\nu}_{k,0} \leq \nu_k$ for each k . In Section 7 we present a simple (and efficient) algorithm that relaxes the need for the lower bounds $\underline{\nu}_{k,0}$.

Remark 1. Later (in Lemma 6) we will show that with high probability $1 \leq w_{k,s} \leq O(s)$. By definition the confidence bounds $\underline{\nu}_{k,t}$ and $\bar{\nu}_{k,t}$ are non-decreasing/increasing respectively. These results are sufficient to guarantee that the new algorithm is numerically stable. It is also worth noting that the running time of Algorithm 1 is $O(1)$ per time step, since all sums can be computed incrementally.

4 UPPER BOUNDS ON THE REGRET

The regret of Algorithm 1 depends in a subtle way on the parameters ν_k . There are four natural cases, which will appear in our main result.

Case 1: Insufficient budget for any jobs. In this case $\ell = 0$ and the optimal algorithm allocates all available resources to the easiest task, which means $M_1^* = 1$. Knowing that $\ell = 0$, the problem can be reduced to a K -armed Bernoulli bandit by restricting the action space to $M_{k,t} = 1$ for all k . Then a bandit algorithm such as UCB1 [Auer et al.,

Algorithm 1 Optimistic Allocation Algorithm

```

1: input:  $n, K, \{\underline{\nu}_{k,0}\}_{k=1}^K$ 
2:  $\delta \leftarrow (nK)^{-2}$  and  $\bar{\nu}_{k,0} = \infty$  for each  $k$ 
3: for  $t \in 1, \dots, n$  do
4:   /* Optimistically choose  $M_{k,t}$  using  $\underline{\nu}_{k,t-1}$  */
5:    $(\forall k \in 1, \dots, K)$  initialise  $M_{k,t} \leftarrow 0$ 
6:   for  $i \in 1, \dots, K$  do
7:      $k \leftarrow \arg \min_{k: M_{k,t}=0} \underline{\nu}_{k,t-1}$ 
8:      $M_{k,t} \leftarrow \min \left\{ \underline{\nu}_{k,t-1}, 1 - \sum_{j=1}^K M_{j,t} \right\}$ 
9:   end for
10:   $(\forall k \in 1, \dots, K)$  observe  $X_{k,t}$ 
11:   $(\forall k \in 1, \dots, K)$  compute weighted estimates:
```

$$w_{k,t} \leftarrow \frac{1}{1 - \frac{M_{k,t}}{\bar{\nu}_{k,t-1}}} \quad \frac{1}{\hat{\nu}_{k,t}} \leftarrow \frac{\sum_{s=1}^t w_{k,s} X_{k,s}}{\sum_{s=1}^t w_{k,s} M_{k,s}}$$

```

12:   $(\forall k \in 1, \dots, K)$  update confidence intervals:
```

$$R_{k,t} \leftarrow \max_{s \leq t} w_{k,s} \quad \hat{V}_{k,t}^2 \leftarrow \sum_{s \leq t} \frac{w_{k,s} M_{k,s}}{\underline{\nu}_{k,t-1}}$$

$$\tilde{\epsilon}_{k,t} \leftarrow \frac{f(R_{k,t}, \hat{V}_{k,t}^2, \delta)}{\sum_{s=1}^t w_{k,s} M_{k,s}}$$

$$\frac{1}{\underline{\nu}_{k,t}} \leftarrow \min \left\{ \frac{1}{\underline{\nu}_{k,t-1}}, \frac{1}{\hat{\nu}_{k,t}} + \tilde{\epsilon}_{k,t} \right\}$$

$$\frac{1}{\bar{\nu}_{k,t}} \leftarrow \max \left\{ \frac{1}{\bar{\nu}_{k,t-1}}, \frac{1}{\hat{\nu}_{k,t}} - \tilde{\epsilon}_{k,t} \right\}$$

```

13: end for
```

```

14: function  $f(R, V^2, \delta)$ 
```

```

15:    $\delta_0 \leftarrow \frac{\delta}{3(R+1)^2(V^2+1)^2}$ 
```

```

16:   return  $\frac{R+1}{3} \log \frac{2}{\delta_0}$ 
       $+ \sqrt{2(V^2+1) \log \frac{2}{\delta_0} + \left(\frac{R+1}{3}\right)^2 \log^2 \frac{2}{\delta_0}}$ 
```

```

17: end function
```

2002] will achieve logarithmic (problem dependent) regret with some dependence on the gaps $\Delta_{1,k} = \frac{1}{\nu_1} - \frac{1}{\nu_k}$. In particular, the regret looks like $R_n \in O\left(\sum_{k=2}^K \frac{\log n}{\Delta_{1,k}}\right)$.

Case 2: Sufficient budget for all jobs. In this case $\ell = K$ and the optimal policy assigns $M_{k,t} = \nu_k$ for all k , which enjoys a reward of K at each time-step. Now Algorithm 1 will choose $M_{k,t} = \underline{\nu}_{k,t-1}$ for all time-steps and by Theorem 4 stated below we will have $\underline{\nu}_{k,t-1}/\nu_k \in O(1 - \frac{1}{t} \log n)$. Consequently, the regret may be bounded by $R_n \in O(\log^2 n)$ with no dependence on the gaps.

Case 3: Sufficient budget for all but one job. Now the algorithm must learn which jobs should be fully allocated. This introduces a weak dependence on the gaps $\Delta_{\ell,k}$ for

$k > \ell$, but choosing the overflow job is trivial. Again we expect the regret to be $O(\log^2 n)$, but with an additional modest dependence on the gaps.

Case 4: General case. In the completely general case even the choice of the overflow job is non-trivial. Ultimately it turns out that in this setting the problem decomposes into two sub-problems. Choosing the jobs to fully allocate, and choosing the overflow job. The first component is fast, since we can make use of the faster learning when fully allocating. Choosing the overflow reduces to the bandit problem as described in case 1.

Our main result is the following theorem bounding the regret of our algorithm.

Theorem 2. *Let δ be as in the algorithm, $\eta_k = \min\{1, \nu_k\} / \nu_{k,0}$, $\tilde{\delta}_k = \frac{\delta}{48\eta_k^4 n^6}$, $c_{k,1} = 27 \log \frac{2}{\delta_k}$, $c_{k,2} = 6 \log \frac{2}{\delta_k}$, $u_{k,j} = \frac{c_{k,1}}{\nu_{k,0} \Delta_{j,k}}$. Then Algorithm 1 suffers regret at most*

$$R_n \leq 1 + \sum_{k=1}^{\ell} c_{k,1} \eta_k (1 + \log n) + \mathbb{1}\{\ell < K\} \left[\sum_{k=\ell+2}^K \frac{c_{k,2}}{\nu_{k,0} \Delta_{\ell+1,k}} + \sum_{k=1}^{\ell+1} c_{k,1} \eta_k (1 + \log n) + \sum_{k=\ell+2}^K c_{k,1} \eta_k (1 + \log u_{\ell+1,k}) + \sum_{k=\ell+1}^K c_{k,1} \eta_k (1 + \log u_{\ell,k}) \right].$$

If we assume $\eta_k \in O(1)$ for each k (reasonable as discussed in Section 7), then the regret bound looks like

$$R_n \in O\left(\ell \log^2 n + \sum_{k=\ell+1}^K \left(\log \frac{1}{\nu_k \Delta_{\ell,k}}\right) \log n + \sum_{k=\ell+2}^K \left(\log \frac{1}{\nu_k \Delta_{\ell+1,k}}\right) \log n + \sum_{k=\ell+1}^K \frac{\log n}{\Delta_{\ell+1,k}}\right), \quad (1)$$

where the first term is due to the gap between $\nu_{k,t}$ and ν_k , the second due to discovering which jobs should be fully allocated, while the third and fourth terms are due to mistakes when choosing the overflow job.

The proof is broken into two components. In the first part we tackle the convergence of $\hat{\nu}_{t,k}$ to ν_k and analyse the width of the confidence intervals, which are data-dependent and shrink substantially faster when $M_{k,t}$ is chosen close to ν_k . In the second component we decompose the regret in terms of the width of the confidence intervals. While we avoided large constants in the algorithm itself, in the proof we focus on legibility. Optimising the constants would complicate an already long result.

5 ESTIMATION

We consider a single job with parameter ν and analyse the estimator and confidence intervals used by Algorithm 1.

We start by showing that the confidence intervals contain the truth with high-probability and then analyse the rate at which the intervals shrink as more data is observed. Somewhat surprisingly the rate has a strong dependence on the data with larger allocations leading to faster convergence.

Let $\{\mathcal{F}_t\}_{t=0}^{\infty}$ be a filtration and let M_1, \dots, M_n be a sequence of positive random variables such that M_t is \mathcal{F}_{t-1} -measurable. Define X_t to be sampled from a Bernoulli distribution with parameter $\beta(M_t/\nu)$ for some $\nu \in [\underline{\nu}_0, \infty]$ and assume that X_t is \mathcal{F}_t -measurable. Our goal is to construct a sequence of confidence intervals $\{[\underline{\nu}_t, \bar{\nu}_t]\}_{t=1}^n$ such that $\nu \in [\underline{\nu}_t, \bar{\nu}_t]$ with high probability and $\bar{\nu}_t - \underline{\nu}_t \rightarrow 0$ as fast as possible. We assume a known lower bound $\underline{\nu}_0 \leq \nu$ and define $\bar{\nu}_0 = \infty$. Recall that the estimator used by Algorithm 1 is defined by

$$w_s = \frac{1}{1 - \frac{M_s}{\bar{\nu}_{t-1}}}, \quad \frac{1}{\hat{\nu}_t} = \frac{\sum_{s=1}^t w_s X_s}{\sum_{s=1}^t w_s M_s}.$$

Fix a number $0 < \delta < 1$ and define $\tilde{\varepsilon}_t = f(R_t, \hat{V}_t^2, \delta) / \sum_{s=1}^t w_s M_s$, where the function f is defined in Algorithm 1, $R_t = \max_{s \leq t} w_s$ and $\hat{V}_t^2 = \sum_{s=1}^t \frac{w_s M_s}{\bar{\nu}_{t-1}}$. The lower and upper confidence bounds on ν^{-1} are defined by,

$$\frac{1}{\underline{\nu}_t} = \min \left\{ \frac{1}{\bar{\nu}_{t-1}}, \frac{1}{\hat{\nu}_t} + \tilde{\varepsilon}_t \right\}, \quad \frac{1}{\bar{\nu}_t} = \max \left\{ \frac{1}{\bar{\nu}_{t-1}}, \frac{1}{\hat{\nu}_t} - \tilde{\varepsilon}_t \right\}.$$

We define $\varepsilon_t = \underline{\nu}_t^{-1} - \bar{\nu}_t^{-1}$ to be the (decreasing) width of the confidence interval. Note that both $\underline{\nu}_t$ and $\bar{\nu}_t$ depend on δ , although this dependence is not shown to minimise clutter.

Theorem 3. *If M_s is chosen such that $M_s \leq \nu_{s-1}$ for all s then $\mathbb{P}\{\exists s \leq t \text{ s.t. } \nu \notin [\underline{\nu}_s, \bar{\nu}_s]\} \leq t\delta$ holds for any $0 < \delta < 1$.*

Proof of Theorem 3. Let F_t be the event $F_t = \{\nu \in [\underline{\nu}_t, \bar{\nu}_t]\}$. Note that since $[\underline{\nu}_t, \bar{\nu}_t] \subset [\underline{\nu}_{t-1}, \bar{\nu}_{t-1}] \subset \dots \subset [\underline{\nu}_0, \bar{\nu}_0]$, $F_t \subset F_{t-1} \subset \dots \subset F_0$. Hence, $F_t = \cap_{s \leq t} F_s$ and it suffices to prove that $\mathbb{P}\{F_t^c\} \leq t\delta$.³

Define $Y_s = w_s X_s - \frac{w_s M_s}{\nu}$ and $S_t = \sum_{s=1}^t Y_s$ and $V_t^2 = \sum_{s=1}^t \text{Var}[Y_s | \mathcal{F}_{s-1}]$. We proceed by induction. Assume $\mathbb{P}\{F_{t-1}^c\} \leq (t-1)\delta$, which is trivial for $t = 1$. Now, on F_{t-1} ,

$$\begin{aligned} V_t^2 &\stackrel{(a)}{=} \sum_{s=1}^t \text{Var}[Y_s | \mathcal{F}_{s-1}] \stackrel{(b)}{=} \sum_{s=1}^t \frac{w_s^2 M_s}{\nu} \left(1 - \frac{M_s}{\nu}\right) \\ &\stackrel{(c)}{=} \sum_{s=1}^t \frac{w_s M_s}{\nu} \left(\frac{1 - \frac{M_s}{\nu}}{1 - \frac{M_s}{\bar{\nu}_{s-1}}}\right) \stackrel{(d)}{\leq} \sum_{s=1}^t \frac{w_s M_s}{\nu} \stackrel{(e)}{\leq} \hat{V}_t^2, \end{aligned}$$

³For an event E , we use E^c to denote its complement.

where (a) is the definition of V_t^2 , (b) follows since w_s is \mathcal{F}_{s-1} -measurable, (c) follows by substituting the definition of w_s , (d) and (e) are true since given F_{t-1} we know that $\underline{\nu}_{s-1} \leq \nu \leq \bar{\nu}_{s-1}$. Therefore $f(R_t, V_t^2, \delta) \leq f(R_t, \hat{V}_t^2, \delta)$, which follows since f is monotone increasing in its second argument. Therefore,

$$\begin{aligned} & \mathbb{P} \left\{ \left| \frac{1}{\hat{\nu}_t} - \frac{1}{\nu} \right| \geq \tilde{\varepsilon}_t \wedge F_{t-1} \right\} \\ &= \mathbb{P} \left\{ \left| \frac{\sum_{s=1}^t w_s X_s}{\sum_{s=1}^t w_s M_s} - \frac{1}{\nu} \right| \geq \frac{f(R_t, \hat{V}_t^2, \delta)}{\sum_{s=1}^t w_s M_s} \wedge F_{t-1} \right\} \\ &\leq \mathbb{P} \left\{ \left| \sum_{s=1}^t w_s X_s - \sum_{s=1}^t \frac{w_s M_s}{\nu} \right| \geq f(R_t, V_t^2, \delta) \wedge F_{t-1} \right\} \\ &= \mathbb{P} \{ |S_t| \geq f(R_t, V_t^2, \delta) \wedge F_{t-1} \}. \end{aligned} \quad (2)$$

By the union bound we have

$$\begin{aligned} & \mathbb{P} \{ |S_t| \geq f(R_t, \hat{V}_t^2, \delta) \vee F_{t-1}^c \} \\ &\leq \mathbb{P} \{ |S_t| \geq f(R_t, V_t^2, \delta) \wedge F_{t-1} \} + \mathbb{P} \{ F_{t-1}^c \} \\ &\stackrel{(a)}{\leq} \delta + \mathbb{P} \{ F_{t-1}^c \} \leq \delta + (t-1)\delta = t\delta, \end{aligned}$$

where (a) follows from a martingale version of Bernstein's inequality adapted from [Bernstein 1946](#) and [Freedman 1975](#). See the supplementary material for details. Therefore $\mathbb{P} \{ |S_t| \leq f(R_t, V_t^2, \delta) \wedge F_{t-1} \} \geq 1 - t\delta$ and so with probability at least $1 - t\delta$ we have that F_{t-1} and

$$\left| \frac{1}{\hat{\nu}_t} - \frac{1}{\nu} \right| \leq \frac{f(R_t, \hat{V}_t^2, \delta)}{\sum_{s=1}^t w_s M_s} = \tilde{\varepsilon}_t,$$

in which case

$$\frac{1}{\underline{\nu}_t} = \min \left\{ \frac{1}{\underline{\nu}_{t-1}}, \frac{1}{\hat{\nu}_t} + \tilde{\varepsilon}_t \right\} \geq \frac{1}{\nu},$$

and similarly $\frac{1}{\bar{\nu}_t} \leq \frac{1}{\nu}$, which implies F_t . Therefore $\mathbb{P} \{ F_t^c \} \leq t\delta$ as required. \square

We now analyse the width $\varepsilon_t \equiv \underline{\nu}_t^{-1} - \bar{\nu}_t^{-1}$ of the confidence interval obtained after t samples are observed. We say that a job is *fully allocated* at time-step s if $M_s = \underline{\nu}_{s-1}$. The first theorem shows that the width ε_t drops with order $O(1/T(t))$, where $T(t) = \sum_{s=1}^t \mathbb{1}\{M_s = \underline{\nu}_{s-1}\}$ is the number of fully allocated time-steps. The second theorem shows that for any $\alpha > 0$, the width ε_t drops with order $O(\sqrt{1/(\alpha U_\alpha(t))})$, where $U_\alpha(t) = \sum_{s=1}^t \mathbb{1}\{M_s \geq \alpha\}$. The dramatic difference in speeds is due to the low variance $\text{Var}[X_t | \mathcal{F}_{t-1}]$ when M_t is chosen close to ν . For the next results define $\eta = \min \{1, \nu\} / \nu_0$ and $\tilde{\delta} = \frac{\delta}{48\eta^4 n^6}$.

Theorem 4. $\varepsilon_t \leq \frac{c_1}{\nu_0(T(t) + 1)}$ where $c_1 = 27 \log \frac{2}{\delta}$.

Theorem 5. $\varepsilon_t \leq \sqrt{\frac{c_2}{\alpha \nu_0 U_\alpha(t)}}$ where $c_2 = 6 \log \frac{2}{\delta}$.

The proofs are based on the following lemma that collects some simple observations:

Lemma 6. *The following hold for any $t \geq 1$:*

1. $w_t M_t \leq \frac{1}{\varepsilon_{t-1}}$, with equality if $M_t = \underline{\nu}_{t-1}$.
2. $1 \leq R_t \leq \frac{1}{\underline{\nu}_0 \varepsilon_{t-1}}$.
3. $\varepsilon_t \geq \frac{1}{t \min \{1, \nu\}}$.
4. $1 - \frac{\underline{\nu}_t}{\nu} \leq \underline{\nu}_t \varepsilon_t$.

Proof. Using the definition of w_s and the fact that M_s is always chosen to be smaller or equal to $\underline{\nu}_{s-1}$, we get

$$w_s \equiv \left(1 - \frac{M_s}{\underline{\nu}_{s-1}}\right)^{-1} \stackrel{(a)}{\leq} \left(1 - \frac{\underline{\nu}_{s-1}}{\bar{\nu}_{s-1}}\right)^{-1} = \frac{1}{\varepsilon_{s-1} \underline{\nu}_{s-1}}.$$

The first claim follows since the inequality (a) can be replaced by equality if $M_s = \underline{\nu}_{s-1}$. The second follows from the definition of R_t and the facts that $(\varepsilon_s)_s$ is non-increasing and $(\underline{\nu}_s)_s$ is non-decreasing. For the third claim we recall that $R_t = \max_{s \leq t} w_s$ and $M_s \leq \nu$. Therefore,

$$\begin{aligned} \varepsilon_t &\stackrel{(a)}{\geq} \min \left\{ \varepsilon_{t-1}, \frac{R_t}{\sum_{s=1}^t w_s M_s} \right\} \\ &\stackrel{(b)}{\geq} \min \left\{ \varepsilon_{t-1}, \frac{1}{t \min \{1, \nu\}} \right\}, \end{aligned}$$

where (a) follows from the definition of ε_t and naive bounding of the function f , (b) follows since $R_t \geq w_s$ for all $s \leq t$ and because $M_s \leq \min \{1, \nu\}$ for all s . Trivial induction and the fact that $\varepsilon_0 = \underline{\nu}_0^{-1} \geq \nu^{-1}$ completes the proof of the third claim. For the final claim we use the facts that $\underline{\nu}_t^{-1} \leq \nu^{-1} + \varepsilon_t$. Therefore, $1 - \frac{\underline{\nu}_t}{\nu} = \underline{\nu}_t \left(\frac{1}{\underline{\nu}_t} - \frac{1}{\nu} \right) \leq \underline{\nu}_t \varepsilon_t$. \square

Lemma 7. $\varepsilon_t \leq \frac{6R_t \log \frac{2}{\delta}}{\sum_{s=1}^t w_s M_s} + \sqrt{\frac{8 \log \frac{2}{\delta}}{\nu_0 \sum_{s=1}^t w_s M_s}}$.

Proof. Let $\delta_t = \delta / (3(R_t + 1)^2 (\hat{V}_t^2 + 1)^2) < 1$. By the definition of ε_t ,

$$\begin{aligned} \varepsilon_t &\leq \frac{2f(R_t, \hat{V}_t^2, \delta)}{\sum_{s=1}^t w_s M_s} \\ &\stackrel{(a)}{\leq} \frac{\frac{4(R_t+1)}{3} \log \frac{2}{\delta_t} + 2\sqrt{2(\hat{V}_t^2 + 1) \log \frac{2}{\delta_t}}}{\sum_{s=1}^t w_s M_s} \\ &\stackrel{(b)}{\leq} \frac{6R_t \log \frac{2}{\delta_t} + \sqrt{\frac{8}{\nu_0} \sum_{s=1}^t w_s M_s \log \frac{2}{\delta_t}}}{\sum_{s=1}^t w_s M_s} \\ &= \frac{6R_t \log \frac{2}{\delta_t}}{\sum_{s=1}^t w_s M_s} + \sqrt{\frac{8 \log \frac{2}{\delta_t}}{\nu_0 \sum_{s=1}^t w_s M_s}}, \end{aligned}$$

where in (a) we used the definition of f , in (b) we substituted the definition of \hat{V}_t^2 and used the facts that $R_t \geq 1$

and $\nu_0 \leq \nu_{t-1}$ and we also used a naive bound. The proof is completed by proving $2/\delta_t \leq 2/\tilde{\delta}$. Indeed, by Lemma 6, $1 \leq R_t \leq \frac{1}{\varepsilon_{t-1}\nu_0} \leq \frac{1}{\varepsilon_t\nu_0}$. We also have $\hat{V}_t^2 \leq tR_t^2$. Thus,

$$\frac{2}{\delta_t} = \frac{6(R_t + 1)^2(\hat{V}_t^2 + 1)^2}{\delta} \leq \frac{6}{\delta} \left(\frac{16t^2}{(\varepsilon_t\nu_0)^4} \right) \stackrel{(a)}{\leq} \frac{2}{\tilde{\delta}},$$

where in (a) we used Lemma 6(3). \square

Proof of Theorem 4. By Lemma 7,

$$\varepsilon_t \leq \frac{6R_t \log \frac{2}{\tilde{\delta}}}{\sum_{s=1}^t w_s M_s} + \sqrt{\frac{8}{\nu_0 \sum_{s=1}^t w_s M_s}} \log \frac{2}{\tilde{\delta}}. \quad (3)$$

We proceed by induction. Assume that $\varepsilon_{s-1} \leq \frac{c_1}{\nu_0(T(s-1)+1)}$, which is trivial for $s = 1$. By Lemma 6(1),

$$\sum_{s=1}^t w_s M_s \geq \sum_{s=1}^{T(t)} \frac{s\nu_0}{c_1} = \frac{\nu_0 T(t)(T(t)+1)}{2c_1}. \quad (4)$$

Therefore,

$$\sqrt{\frac{8}{\nu_0 \sum_{s=1}^t w_s M_s}} \log \frac{2}{\tilde{\delta}} \stackrel{(a)}{\leq} \frac{1}{\nu_0 T(t)} \sqrt{4c_1 \log \frac{2}{\tilde{\delta}}}. \quad (5)$$

Now we work on the first term in (3). If $\varepsilon_{t-1} \leq \frac{c_1}{\nu_0(T(t)+1)}$, then we are done, since ε_s is non-increasing. Otherwise, we use Lemma 6(2) to obtain,

$$\begin{aligned} \frac{6R_t}{\sum_{s=1}^t w_s M_s} \log \frac{2}{\tilde{\delta}} &\leq \frac{6}{\nu_0 \varepsilon_{t-1} \sum_{s=1}^t w_s M_s} \log \frac{2}{\tilde{\delta}} \\ &\stackrel{(a)}{\leq} \frac{3}{\nu_0 T(t)} \log \frac{2}{\tilde{\delta}}, \end{aligned} \quad (6)$$

where in (a) we used (4) and the lower bound on ε_{t-1} . Substituting (5) and (6) into (3) we have

$$\varepsilon_t \leq \frac{1}{\nu_0 T(t)} \sqrt{4c_1 \log \frac{2}{\tilde{\delta}}} + \frac{3}{\nu_0 T(t)} \log \frac{2}{\tilde{\delta}}.$$

Choosing $c_1 = 27 \log \frac{2}{\tilde{\delta}}$ leads to

$$\begin{aligned} \varepsilon_t &\leq \frac{1}{\nu_0 T(t)} \sqrt{4 \cdot 27 \log^2 \frac{2}{\tilde{\delta}}} + \frac{3}{\nu_0 T(t)} \log \frac{2}{\tilde{\delta}} \\ &\leq \frac{27}{\nu_0 (T(t)+1)} \log \frac{2}{\tilde{\delta}} = \frac{c_1}{\nu_0 (T(t)+1)}, \end{aligned}$$

which completes the induction and proof. \square

Proof of Theorem 5. Firstly, by Lemma 7,

$$\varepsilon_t \leq \frac{6R_t}{\sum_{s=1}^t w_s M_s} \log \frac{2}{\tilde{\delta}} + \sqrt{\frac{8}{\nu_0 \sum_{s=1}^t w_s M_s}} \log \frac{2}{\tilde{\delta}}.$$

The second term is easily bounded by using the fact that $w_s \geq 1$ and the definition of $U_\alpha(t)$,

$$\sqrt{\frac{8}{\nu_0 \sum_{s=1}^t w_s M_s}} \log \frac{2}{\tilde{\delta}} \leq \sqrt{\frac{8}{\nu_0 U_\alpha(t) \alpha}} \log \frac{2}{\tilde{\delta}}.$$

For the first term we assume $\varepsilon_{t-1} \geq \sqrt{\frac{c_2}{\nu_0 U_\alpha(t) \alpha}}$, since otherwise we can apply monotonicity of ε_t . Therefore

$$\begin{aligned} \frac{6R_t}{\sum_{s=1}^t w_s M_s} \log \frac{2}{\tilde{\delta}} &\leq \frac{6}{\nu_0 \varepsilon_{t-1} \sum_{s=1}^t w_s M_s} \log \frac{2}{\tilde{\delta}} \\ &\leq \sqrt{\frac{U_\alpha(t) \alpha \nu_0}{c_2}} \cdot \frac{6 \log \frac{2}{\tilde{\delta}}}{\nu_0 U_\alpha(t) \alpha} \leq 6 \sqrt{\frac{1}{c_2 \alpha \nu_0 U_\alpha(t)}} \log \frac{2}{\tilde{\delta}}. \end{aligned}$$

Now choose $c_2 = 6 \log \frac{2}{\tilde{\delta}}$ to complete the result. \square

6 PROOF OF THEOREM 2

We are now ready to use the results of Section 5 to bound the regret of Algorithm 1. The first step is to decompose the regret into two cases depending on whether or not the confidence intervals contain the truth. The probability that they do not is low, so this contributes negligibly to the regret. When the confidence intervals are valid we break the problem into two components. The first is the selection of the processes to fully allocation, which leads to the $O(\log^2 n)$ part of the bound. The second component involves analysing the selection of the overflow process, where the approach is reminiscent of the analysis for the UCB algorithm for stochastic bandits [Auer et al., 2002].

Let $F_{k,t}$ denote the event when none of the confidence intervals underlying job k fail up to time t :

$$F_{k,t} = \{\forall s \leq t : \nu \in [\underline{\nu}_{k,s}, \bar{\nu}_{k,s}]\}.$$

The algorithm uses $\delta = (nK)^{-2}$, which is sufficient by a union bound and Theorem 3 to ensure that,

$$\mathbb{P}\{G^c\} \leq \frac{1}{nK}, \quad \text{where } G = \bigcap_{k=1}^K F_{k,n}. \quad (7)$$

The regret can be decomposed into two cases depending on whether G holds:

$$\begin{aligned} R_n &= \mathbb{E} \sum_{t=1}^n r_t \stackrel{(a)}{=} \mathbb{E} \mathbb{1}\{G^c\} \sum_{t=1}^n r_t + \mathbb{E} \mathbb{1}\{G\} \sum_{t=1}^n r_t \quad (8) \\ &\stackrel{(b)}{\leq} \mathbb{E} \mathbb{1}\{G^c\} nK + \mathbb{E} \mathbb{1}\{G\} \sum_{t=1}^n r_t \stackrel{(c)}{\leq} 1 + \mathbb{E} \mathbb{1}\{G\} \sum_{t=1}^n r_t, \end{aligned}$$

where (a) follows from the definition of expectation, (b) is true by bounding $r_t \leq K$ for all t , and (c) follows from (7). For the remainder we assume G holds and use Theorems 4

and 5 combined with the definition of the algorithm to control the second term in (8). The first step is to decompose the regret in round t :

$$r_t = \ell^* + \frac{S^*}{\nu_{\ell+1}} - \sum_{k=1}^K \beta \left(\frac{M_{k,t}}{\nu_k} \right).$$

By the assumption that G holds we know for all $t \leq n$ and k that $\bar{\nu}_{k,t}^{-1} \leq \nu_k^{-1} \leq \nu_{k,t}^{-1}$. Therefore $M_{k,t} \leq \nu_{k,t-1} \leq \nu_k$, which means that $\beta(M_{k,t}/\nu_k) = M_{k,t}/\nu_k$. Define $\pi_t(i) \in \{1, \dots, K\}$ such that $\nu_{\pi_t(i),t-1} \leq \nu_{\pi_t(i+1),t-1}$. Also let

$$\begin{aligned} A_t &= \{k : M_{k,t} = \nu_{k,t-1}\}, \\ A_t^{\leq j} &= A_t \cap \{\pi_i(t) : 1 \leq i \leq j\}, \\ T_k(t) &= \sum_{s=1}^t \mathbb{1}\{k \in A_s\} \quad \text{and} \quad B_t = \pi_t(\ell+1). \end{aligned}$$

Informally, A_t is the set of jobs that are fully allocated at time-step t , $A_t^{\leq j}$ is a subset of A_t containing the j jobs believed to be easiest, $T_k(t)$ is the number of times job k has been fully allocated at time-step t , and B_t is the $(\ell+1)$ th easiest job at time-step t (this is only defined if $\ell < K$ and will only be used in that case).

Lemma 8. For all t , $|A_t| \geq \ell$ and if $|A_t| = \ell$, then $M_{B_t,t} \geq S^*$.

Proof. $|A_t| = \max \left\{ j : \sum_{i=1}^j \nu_{\pi_t(i),t-1} \leq 1 \right\}$. But $\nu_{k,t-1} \leq \nu_k$ for all k and t , so $\sum_{i=1}^{\ell} \nu_{\pi_t(i),t-1} \leq \sum_{k=1}^{\ell} \nu_{k,t-1} \leq \sum_{k=1}^{\ell} \nu_k \leq 1$. Therefore $|A_t| \geq \ell$. If $|A_t| = \ell$, then $B_t \notin A_t$ is the overflow job and so $M_{B_t,t} = 1 - \sum_{k \in A_t} \nu_{k,t-1} \geq 1 - \sum_{k \in A^*} \nu_{k,t-1} \geq 1 - \sum_{k \in A^*} \nu_k \equiv S^*$. \square

We now decompose the regret, while still assuming that G holds:

$$\begin{aligned} \sum_{t=1}^n r_t &= \sum_{t=1}^n \left(\ell + \frac{S^*}{\nu_{\ell+1}} - \sum_{k=1}^K \frac{M_{k,t}}{\nu_k} \right) \\ &\leq \sum_{t=1}^n \sum_{k \in A_t^{\leq \ell}} \left(1 - \frac{M_{k,t}}{\nu_k} \right) \\ &\quad + \mathbb{1}\{\ell < K\} \sum_{t=1}^n \left(\frac{S^*}{\nu_{\ell+1}} - \frac{M_{B_t,t}}{\nu_{B_t}} \right). \end{aligned} \quad (9)$$

Let us bound the first sum:

$$\begin{aligned} \sum_{t=1}^n \sum_{k \in A_t^{\leq \ell}} \left(1 - \frac{M_{k,t}}{\nu_k} \right) &= \sum_{t=1}^n \sum_{k=1}^K \mathbb{1}\{k \in A_t^{\leq \ell}\} \left(1 - \frac{\nu_{k,t-1}}{\nu_k} \right) \end{aligned}$$

$$\begin{aligned} &\stackrel{(a)}{\leq} \sum_{t=1}^n \sum_{k=1}^K \mathbb{1}\{k \in A_t^{\leq \ell}\} \nu_{k,t-1} \varepsilon_{k,t-1} \\ &\stackrel{(b)}{\leq} \sum_{t=1}^n \sum_{k=1}^K \mathbb{1}\{k \in A_t^{\leq \ell}\} \frac{c_{k,1} \nu_{k,t-1}}{\nu_{k,0} T_k(t)}, \end{aligned} \quad (11)$$

where (a) follows by Lemma 6 and (b) by Theorem 4.

Lemma 9. If $k > j$, then

$$\sum_{t=1}^n \mathbb{1}\{k \in A_t^{\leq j}\} \leq \frac{c_{k,1}}{\nu_{k,0} \Delta_{j,k}} =: u_{j,k}.$$

Proof. Assume $k \in A_t^{\leq j}$. Therefore $\nu_{k,t-1} \leq \nu_j$. But if $u_{j,k} < \sum_{s=1}^t \mathbb{1}\{k \in A_s^{\leq j}\} \leq T_k(t-1) + 1$, then

$$\begin{aligned} \frac{1}{\nu_{k,t-1}} &\leq \frac{1}{\nu_k} + \varepsilon_{k,t-1} = \frac{1}{\nu_j} + \varepsilon_{k,t-1} - \Delta_{j,k} \\ &\stackrel{(a)}{\leq} \frac{1}{\nu_j} + \frac{c_{k,1}}{\nu_{k,0} (T_k(t-1) + 1)} - \Delta_{j,k} < \frac{1}{\nu_j}, \end{aligned}$$

where (a) follows from Theorem 4. Therefore $k \in A_t^{\leq j}$ implies that $\sum_{s=1}^t \mathbb{1}\{k \in A_s^{\leq j}\} \leq u_{j,k}$ and so $\sum_{t=1}^n \mathbb{1}\{k \in A_t^{\leq j}\} \leq u_{j,k}$ as required. \square

Continuing (11) by applying Lemma 9 with $j = \ell$:

$$\begin{aligned} &\sum_{t=1}^n \sum_{k=1}^K \mathbb{1}\{k \in A_t^{\leq \ell}\} \frac{c_{k,1} \nu_{k,t-1}}{\nu_{k,0} T_k(t)} \\ &= \sum_{t=1}^n \sum_{k \in A^*} \mathbb{1}\{k \in A_t^{\leq \ell}\} \frac{c_{k,1} \nu_{k,t-1}}{\nu_{k,0} T_k(t)} \\ &\quad + \sum_{t=1}^n \sum_{k \notin A^*} \mathbb{1}\{k \in A_t^{\leq \ell}\} \frac{c_{k,1} \nu_{k,t-1}}{\nu_{k,0} T_k(t)} \quad (12) \\ &\stackrel{(a)}{\leq} \sum_{k \in A^*} \sum_{t=1}^n \frac{c_{k,1} \eta_k}{t} + \sum_{k \notin A^*} \sum_{t=1}^{u_{\ell,k}} \frac{c_{k,1} \eta_k}{t} \\ &\leq \sum_{k=1}^{\ell} c_{k,1} \eta_k (1 + \log n) + \sum_{k=\ell+1}^K c_{k,1} \eta_k (1 + \log u_{\ell,k}), \end{aligned}$$

where (a) follows by Lemma 9 and the fact that $k \in A_t^{\leq \ell}$ implies that $\frac{\nu_{k,t-1}}{\nu_{k,0}} \leq \eta_k$. Now if $\ell = K$, then the second term in (9) is zero and the proof is completed by substituting the above result into (9) and then into (8). So now we assume $\ell > K$ and bound the second term in (9) as follows:

$$\begin{aligned} \sum_{t=1}^n \left(\frac{S^*}{\nu_{\ell+1}} - \frac{M_{B_t,t}}{\nu_{B_t}} \right) &\leq \sum_{t=1}^n \mathbb{1}\{B_t \in A_t\} \left(1 - \frac{\nu_{B_t,t-1}}{\nu_{B_t}} \right) \\ &\quad + \sum_{t=1}^n \mathbb{1}\{B_t \notin A_t\} \left(\frac{S^*}{\nu_{\ell+1}} - \frac{S^*}{\nu_{B_t}} \right), \end{aligned} \quad (13)$$

where we used Lemma 8 and $S^* \leq 1$ and that if $B_t \in A_t$, then $M_{B_t,t} = \underline{\nu}_{B_t,t-1}$. Bounding each term separately:

$$\begin{aligned}
& \sum_{t=1}^n \mathbb{1}\{B_t \in A_t\} \left(1 - \frac{\underline{\nu}_{B_t,t-1}}{\nu_{B_t}}\right) \\
\stackrel{(a)}{\leq} & \sum_{k=1}^K \sum_{t=1}^n \mathbb{1}\{k \in A_t^{\leq \ell+1}\} \left(1 - \frac{\underline{\nu}_{k,t-1}}{\nu_k}\right) \\
\stackrel{(b)}{\leq} & \sum_{k=1}^K \sum_{t=1}^n \mathbb{1}\{k \in A_t^{\leq \ell+1}\} \underline{\nu}_{k,t-1} \varepsilon_{k,t-1} \\
\stackrel{(c)}{\leq} & \sum_{k=1}^K \sum_{t=1}^n \mathbb{1}\{k \in A_t^{\leq \ell+1}\} \frac{c_{k,1} \underline{\nu}_{k,t-1}}{\underline{\nu}_{k,0} T_k(t)} \\
\stackrel{(d)}{\leq} & \sum_{k=1}^{\ell+1} c_{k,1} \eta_k (1 + \log n) + \sum_{k=\ell+2}^K c_{k,1} \eta_k (1 + \log u_{\ell+1,k}),
\end{aligned} \tag{14}$$

where (a) follows since $B_t \in A_t$ implies that $B_t \in A_t^{\leq \ell+1}$, (b) follows from Lemma 6(4), (c) by Theorem 4, and (d) follows from Lemma 9 and the same analysis as (12). For the second term we need the following lemma, which uses Theorem 5 and a reasoning analogous to that of Auer et al. [2002] to bound the regret of the UCB algorithm for stochastic bandits:

Lemma 10. Let $U_k(t) = \sum_{s=1}^t \mathbb{1}\{M_{k,s} \geq S^*\}$ and $k > \ell + 1$. If $U_k(t) \geq \frac{c_{k,2}}{S^* \underline{\nu}_{k,0} \Delta_{\ell+1,k}^2} =: v_k$, then $k \neq B_t$.

Proof. If $\underline{\nu}_{k,t-1} > \nu_{\ell+1}$, then $k \neq B_t$. Furthermore, if $U_k(t) > v_k$, then

$$\begin{aligned}
\frac{1}{\underline{\nu}_{k,t-1}} & \leq \frac{1}{\nu_k} + \varepsilon_{k,t-1} = \frac{1}{\nu_{\ell+1}} - \Delta_{\ell+1,k} + \varepsilon_{k,t-1} \\
& \stackrel{(a)}{\leq} \frac{1}{\nu_{\ell+1}} - \Delta_{\ell+1,k} + \sqrt{\frac{c_{k,2}}{\underline{\nu}_{k,0} S^* U_k(t)}} < \frac{1}{\nu_{\ell+1}},
\end{aligned}$$

where (a) follows from Theorem 5. \square

Therefore

$$\begin{aligned}
& \sum_{t=1}^n \mathbb{1}\{B_t \notin A_t\} \left(\frac{S^*}{\nu_{\ell+1}} - \frac{S^*}{\nu_{B_t}}\right) \\
\stackrel{(a)}{\leq} & S^* \sum_{k=1}^K \sum_{t=1}^n \mathbb{1}\{k = B_t \notin A_t\} \Delta_{\ell+1,k} \\
\stackrel{(b)}{\leq} & S^* \sum_{k=\ell+2}^K \sum_{t=1}^n \mathbb{1}\{k = B_t \notin A_t\} \Delta_{\ell+1,k} \\
\stackrel{(c)}{\leq} & S^* \sum_{k=\ell+2}^K \sum_{t=1}^n \mathbb{1}\{k = B_t \wedge M_{k,t} \geq S^*\} \Delta_{\ell+1,k} \\
\stackrel{(d)}{\leq} & \sum_{k=\ell+2}^K S^* \Delta_{\ell+1,k} v_k \stackrel{(e)}{=} \sum_{k=\ell+2}^K \frac{c_{k,2}}{\underline{\nu}_{k,0} \Delta_{\ell+1,k}}, \tag{15}
\end{aligned}$$

where (a) follows from the definition of $\Delta_{\ell+1,k}$ and the fact that if $B_t \notin A_t$, then $|A_t| = \ell$, (b) follows since $\Delta_{\ell+1,k}$ is negative for $k \leq \ell + 1$, (c) by Lemma 8, (d) by Lemma 10, and (e) by the definition of v_k . Substituting (14) and (15) into (13) we have

$$\begin{aligned}
& \sum_{t=1}^n \left(\frac{S^*}{\nu_{\ell+1}} - \frac{M_{B_t,t}}{\nu_{B_t}}\right) \leq \sum_{k=1}^{\ell+1} c_{k,1} \eta_k (1 + \log n) \\
& + \sum_{k=\ell+2}^K c_{k,1} \eta_k (1 + \log u_{\ell+1,k}) + \sum_{k=\ell+2}^K \frac{c_{k,2}}{\underline{\nu}_{k,0} \Delta_{\ell+1,k}}.
\end{aligned}$$

We then substitute this along with (12) into (9) and then (8) to obtain

$$\begin{aligned}
R_n & \leq 1 + \sum_{k=1}^{\ell} c_{k,1} \eta_k (1 + \log n) \\
& + \mathbb{1}\{\ell < K\} \left[\sum_{k=\ell+2}^K \frac{c_{k,2}}{\underline{\nu}_{k,0} \Delta_{\ell+1,k}} + \sum_{k=1}^{\ell+1} c_{k,1} \eta_k (1 + \log n) \right. \\
& \left. + \sum_{k=\ell+2}^K c_{k,1} \eta_k (1 + \log u_{\ell+1,k}) + \sum_{k=\ell+1}^K c_{k,1} \eta_k (1 + \log u_{\ell,k}) \right].
\end{aligned}$$

7 INITIALISATION

Previously we assumed a known lower bound $\underline{\nu}_{k,0} \leq \nu_k$ for each k . In this section we show that these bounds are easily obtained using a halving trick. In particular, the following algorithm computes a lower bound $\underline{\nu}_0 \leq \nu$ for a single job with unknown parameter ν .

Algorithm 2 Initialisation of $\underline{\nu}_0$

- 1: **for** $t = 1, \dots, \infty$ **do**
 - 2: Allocate $M_t = 2^{-t}$ and observe X_t
 - 3: **if** $X_t = 0$ **then return** $\underline{\nu}_0 \leftarrow 2^{-t}$.
 - 4: **end for**
-

A naive way to eliminate the need for the lower bounds $(\underline{\nu}_{k,0})_k$ is simply to run Algorithm 2 for each job sequentially. Then the following proposition (proven in supplementary material) shows that $\eta \in O(1)$ is reasonable, which justifies the claim made in (1) that the η_k terms appearing in Theorem 2 are $O(1)$.

Proposition 11. If $\eta = \frac{\min\{1, \nu\}}{\underline{\nu}_0}$, then $\mathbb{E}\eta \leq 4$.

The problem with the naive method is that the expected running time of Algorithm 2 is $O(\log \frac{1}{\nu})$, which may be arbitrary large for small ν and lead to a high regret *during the initialisation period*. Fortunately, the situation when ν is small is easy to handle, since the amount of resources required to complete such a job is also small. The trick is to run K offset instances of Algorithm 2 alongside a modified version of Algorithm 1. First we describe the parallel

implementations of Algorithm 2. For job k , start Algorithm 2 in time-step k , which means that the total amount of resources used by the parallel copies of Algorithm 2 in time-step t is bounded by

$$\sum_{k=1}^K \mathbb{1}\{t \geq k\} 2^{k-t-1} \leq \min \{1, 2^{K-t}\}. \quad (16)$$

Algorithm 1 is implemented starting from time-step 1, but only allocates resources to jobs for which the initialisation process has completed. Estimates are computed using only the samples for which Algorithm 1 chose the allocation, which ensures that they are based on allocations with $M_{k,t} \leq \nu_k$. Note that the analysis of the modified algorithm does not depend on the order in which the parallel processes are initialised. The regret incurred by the modified algorithm is given in order notation in (1). The proof is omitted, but relies on two observations. First, that the expected number of time-steps that a job is not (at least) fully allocated while it is being initialised is 2. The second is that the resources available to Algorithm 1 at time-step t converges exponentially fast to 1 by (16).

8 MINIMAX LOWER BOUNDS

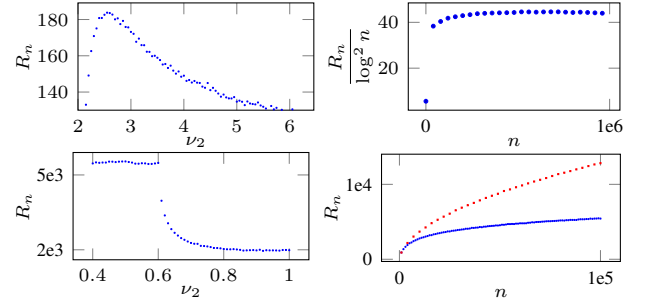
Despite the continuous action space, the techniques used when proving minimax lower bounds for standard stochastic bandits [Auer et al., 1995] can be adapted to our setting. The proof is included in the supplementary material.

Theorem 12. *Given fixed n and $8n \geq K \geq 2$ and an arbitrary algorithm, there exists an allocation problem for which the expected regret satisfies $R_n \geq \frac{\sqrt{nK}}{16\sqrt{2}}$.*

9 EXPERIMENTS

All code and data is available in the supplementary material. Data points were generated using the modified algorithm described in Section 7 and by taking the mean of 300 samples. With this many samples the standard error is relatively low (and omitted for readability). We should note that the variance in the regret of the modified algorithm is reasonably large, because the regret depends linearly on the random η_k . For known lower bounds the variance is extremely low. To illustrate the behaviour of the algorithm we performed four experiments on synthetic data with $K = 2$, which are plotted below as TL (top left), TR, BL, BR (bottom right) respectively. In TL we fixed $n = 10^4$, $\nu_1 = 2$ and plotted the regret as a function of $\nu_2 \in [2, 10]$. The experiment shows the usual bandit-like dependence on the gap $1/\Delta_{1,2}$. In TR we fixed $\nu_1 = 4/10$, $\nu_2 = 6/10$ and plotted $R_n / \log^2 n$ as a function of n . The experiment lies within case 2 described in Section 4 and shows that the algorithm suffers regret $R_n \approx 45 \log^2 n$ as predicted by Theorem 2. In BL we fixed $n = 10^5$, $\nu_1 = 4/10$ and plotted the regret as a function of $\nu_2 \in [4/10, 1]$. The results show

the algorithm suffering $O(\log^2 n)$ regret for both processes until the critical point when $\nu_2 > 6/10$ when the second process can no longer be fully allocated, which is quickly learned and the algorithm suffers $O(\log^2 n)$ regret for only one process. In BR we fixed $\nu_1 = 4/10$ and $\nu_2 = 6/10$ and plotted the regret as a function of n for two algorithms. The first algorithm (solid blue) is the modified version of Algorithm 1 as described in Section 7. The second (dotted red) is the same, but uses the unweighted estimator $w_{k,t} = 1$ for all k and t . The result shows that both algorithms suffer sub-linear regret, but that the weighted estimator is a significant improvement over the unweighted one.



10 CONCLUSIONS

We introduced the linear stochastic resource allocation problem and a new optimistic algorithm for this setting. Our main result shows that the new algorithm enjoys a (squared) logarithmic problem-dependent regret. We also presented a minimax lower bound of $\Omega(\sqrt{nK})$, which is consistent with the problem-dependent upper bound. The simulations confirm the theory and highlight the practical behaviour of the new algorithm. There are many open questions and possibilities for future research. Most important is whether the $\log^2 n$ can be reduced to $\log n$. Problem-dependent lower bounds would be interesting. The algorithm is not anytime (although a doubling trick presumably works in theory). Developing and analysing algorithms when the horizon is not known, and have high-probability bounds are both of interest. We also wonder if Thompson sampling can be efficiently implemented for some reasonable prior, and if it enjoys the same practical and theoretical guarantees in this domain as it does for bandits. Other interesting extensions are when resources are not replenished, or the state of the jobs follow a Markov process. Finally, we want to emphasise that we have made just the first steps towards developing this new and interesting setting. We hope to see significant activity extending and modifying the model/algorithm for specific problems.

Acknowledgements This work was supported by the Alberta Innovates Technology Futures, NSERC, by EU Framework 7 Project No. 248828 (ADVANCE), and by Israeli Science Foundation grant ISF- 1567/10. Part of this work was done while Csaba Szepesvári was visiting Technion.

References

- Rajeev Agrawal, Demosthenis Teneketzis, and Venkatachalam Anantharam. Asymptotically efficient adaptive allocation schemes for controlled i.i.d. processes: Finite parameter space. *IEEE Transaction on Automatic Control*, 34:258–267, 1989.
- Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*, pages 322–331. IEEE, 1995.
- Peter Auer, Nicoló Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256, 2002.
- Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksanders Slivkins. Bandits with knapsacks. In *FOCS*, pages 207–216, 2013.
- Gábor Bartók. A near-optimal algorithm for finite partial-monitoring games against adversarial opponents. In *COLT*, pages 696–710, 2013.
- Gábor Bartók, Dávid Pál, and Csaba Szepesvári. Minimax regret of finite partial-monitoring games in stochastic environments. In *COLT 2011*, pages 133–154, 2011.
- Sergei Bernstein. *The Theory of Probabilities (Russian)*. Moscow, 1946.
- Ramazan Bitirgen, Engin Ipek, and Jose F Martinez. Coordinated management of multiple interacting resources in chip multiprocessors: A machine learning approach. In *Proceedings of the 41st annual IEEE/ACM International Symposium on Microarchitecture*, pages 318–329. IEEE Computer Society, 2008.
- Josef Broder and Paat Rusmevichientong. Dynamic pricing under a general parametric choice model. *Operations Research*, 60(4):965–980, 2012.
- Sébastien Bubeck and Nicolò Cesa-Bianchi. *Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems*. Foundations and Trends in Machine Learning. Now Publishers Incorporated, 2012. ISBN 9781601986269.
- Wenkui Ding, Tao Qin, Xu-Dong Zhang, and Tie-Yan Liu. Multi-armed bandit with budget constraint and variable costs. In *AAAI*, 2013.
- Sarah Filippi, Olivier Cappé, Aurélien Garivier, and Csaba Szepesvári. Parametric bandits: The generalized linear case. In *NIPS*, pages 586–594, December 2010.
- Dean P. Foster and Alexander Rakhlin. No internal regret via neighborhood watch. *Journal of Machine Learning Research - Proceedings Track (AISTATS)*, 22:382–390, 2012.
- David A. Freedman. On tail probabilities for martingales. *The Annals of Probability*, 3(1):100–118, 02 1975.
- András György, Levente Kocsis, Ivett Szabó, and Csaba Szepesvári. Continuous time associative bandit problems. In *IJCAI-07*, pages 830–835, 2007.
- Engin Ipek, Onur Mutlu, José F. Martínez, and Rich Caruana. Self-optimizing memory controllers: A reinforcement learning approach. *SIGARCH Comput. Archit. News*, 36(3):39–50, June 2008. ISSN 0163-5964.
- Tor Lattimore, Koby Crammer, and Csaba Szepesvári. Optimal resource allocation with semi-bandit feedback. *arXiv preprint arXiv:????????*, 2014.
- Chun Liu, Anand Sivasubramaniam, and Mahmut Kandemir. Organizing the last line of defense before hitting the memory wall for cmps. In *Software, IEE Proceedings-*, pages 176–185. IEEE, 2004.
- Daniel Russo and Benjamin Van Roy. Eluder dimension and the sample complexity of optimistic exploration. In *NIPS*, pages 2256–2264, 2013.
- Aldo Rustichini. Minimizing regret: The general case. *Games and Economic Behavior*, 29(1–2):224–243, 1999.
- G Edward Suh, Srinivas Devadas, and Larry Rudolph. A new memory monitoring scheme for memory-aware scheduling and partitioning. In *High-Performance Computer Architecture, 2002. Proceedings. Eighth International Symposium on*, pages 117–128. IEEE, 2002.
- Long Tran-Thanh, Archie C. Chapman, Alex Rogers, and Nicholas R. Jennings. Knapsack based optimal policies for budget-limited multi-armed bandits. In *AAAI*, 2012.

Quantifying Nonlocal Informativeness in High-Dimensional, Loopy Gaussian Graphical Models

Daniel Levine

Lab. for Information and Decision Syst.
Massachusetts Institute of Technology
Cambridge, MA 02139
dlevine@mit.edu

Jonathan P. How

Lab. for Information and Decision Syst.
Massachusetts Institute of Technology
Cambridge, MA 02139
jhow@mit.edu

Abstract

We consider the problem of selecting informative observations in Gaussian graphical models containing both cycles and nuisances. More specifically, we consider the subproblem of quantifying conditional mutual information measures that are *nonlocal* on such graphs. The ability to efficiently quantify the information content of observations is crucial for resource-constrained data acquisition (adaptive sampling) and data processing (active learning) systems. While closed-form expressions for Gaussian mutual information exist, standard linear algebraic techniques, with complexity cubic in the network size, are intractable for high-dimensional distributions. We investigate the use of embedded trees for computing nonlocal pairwise mutual information and demonstrate through numerical simulations that the presented approach achieves a significant reduction in computational cost over inversion-based methods.

1 INTRODUCTION

In resource-constrained inferential settings, uncertainty can be efficiently minimized with respect to a resource budget by acquiring or processing the most informative subset of observations – a problem known as *active inference* (Krause and Guestrin, 2005; Williams et al., 2007). Yet despite the myriad recent advances in both understanding and streamlining inference through probabilistic graphical models (Koller and Friedman, 2009), there does not exist a comparable wealth of knowledge regarding how information measures propagate on these graphs. This paper considers the problem of efficiently quantifying a measure of informativeness across nonlocal pairings in a loopy Gaussian graphical model.

This paper assumes a model has been provided, and the ensuing goal is to interpret relationships within this model

in the context of informativeness. This assumption is motivated by the hypothesis that, regardless of the specific sensing modalities or communication platforms used in an information collection system, the underlying phenomena can be described by *some* stochastic process structured according to a probabilistic graphical model. The sparsity of that model determines the efficiency of inference procedures. In contrast to methods for estimating information measures directly from raw data (e.g., Kraskov et al., 2004), the approach of this paper does not require the prior enumeration of interaction sets that one wishes to quantify, and the presented algorithm computes *conditional* information measures that account for statistical redundancy between observations.

This paper specifically addresses the common issue of nuisances in the model – variables that are not of any extrinsic importance, but act as intermediaries between random variables that are either observable or of inferential interest. Marginalization of nuisances can be both computationally expensive and detrimental to the sparsity of the graph, which, in the interest of efficient model utilization, one wishes to retain. Ignoring nuisances by treating them as relevant can result in observation selectors fixated on reducing uncertainty in irrelevant portions of the underlying distribution (Levine and How, 2013). In terms of information quantification, nuisances can induce nonlocality in the sense that observations and relevant latent variables are not adjacent in the graph, motivating the study of how information measure propagate through graphical models.

In this paper, we investigate the use of embedded trees (Sudderth et al., 2004) for efficiently quantifying nonlocal mutual information in loopy Gaussian graphs. The formal problem statement and a characterization thereof is described in Section 2. Some preliminary material and prior algorithmic technologies are reviewed in Section 3. Our method for quantify mutual information using embedded trees, ET-MIQ, is described in Section 4 and demonstrated through experimental results in Section 5. A discussion of ET-MIQ in comparison to alternative methods and in anticipation of future extensions is provided in Section 6.

2 PROBLEM STATEMENT

Let $\mathbf{x} = (x_1, \dots, x_N)$ be a collection of N random variables (or disjoint subvectors) with joint distribution $p_{\mathbf{x}}(\cdot)$. Let index set $\mathcal{V} = \{1, \dots, N\}$ be partitioned such that $\mathcal{V} = \mathcal{U} \cup \mathcal{S}$, where $\mathcal{U} \subset \mathcal{V}$ indexes latent (unobservable) variables, and where $\mathcal{S} \subset \mathcal{V}$ indexes observable variables, whose realizations $\mathbf{x}_s = x_s, s \in \mathcal{S}$ may be obtained by expending some resource. Let $c : 2^{\mathcal{S}} \rightarrow \mathbb{R}_{\geq 0}$ be the cost function that maps subsets of observable variables to a resource cost, and let $\beta \in \mathbb{R}_{\geq 0}$ be a resource budget. Given a subset $\mathcal{R} \subseteq \mathcal{U}$ of *relevant latent variables*, which are of inferential interest and about which one wishes to reduce uncertainty, the general *focused active inference* problem (Levine and How, 2013) is

$$\begin{aligned} & \text{maximize}_{\mathcal{A} \subseteq \mathcal{S}} && I(\mathbf{x}_{\mathcal{R}}; \mathbf{x}_{\mathcal{A}}) \\ & \text{s.t.} && c(\mathcal{A}) \leq \beta, \end{aligned} \quad (1)$$

where $I(\cdot, \cdot)$ is the mutual information measure (cf. Section 3.3).

It is well known that (1) is **NP**-hard (Ko et al., 1995; Krause and Guestrin, 2009). Despite this, suboptimal heuristics such as greedy selection have been analyzed in the context of submodularity (Nemhauser et al., 1978), leading to various performance bounds (Golovin and Krause, 2010; Krause and Guestrin, 2005; Williams et al., 2007). In the focused case, where there are nuisances $\mathcal{U} \setminus \mathcal{R}$ in the problem, the objective in (1) is in general not submodular (Krause et al., 2008), although online-computable performance bounds can be established through submodular relaxations (Levine and How, 2013).

However, efficient computation of the mutual information objective in (1) has remained elusive for all but simple models – symmetric discrete distributions (Choi et al., 2011) and Gaussian trees (Levine and How, 2013). Just as covariance analysis in the Kalman filtering framework can be used to anticipate the uncertainty evolution in a linear-Gaussian state space model, which is Markov to a minimal tree-shaped Gaussian graph, this paper aims to provide a general *preposterior analysis* of uncertainty reduction in nontree Gaussian systems.

This paper specifically considers the class of Gaussian distributed vectors $\mathbf{x} \sim \mathcal{N}^{-1}(\mathbf{0}, J)$ with inverse covariance matrices J , each of which is Markov to an undirected graph with cycles. For large N , evaluating the MI objective in (1) via matrix inversion is cubic in N , which may be prohibitively expensive. The aim of this paper is explicating an iterative algorithm for computing MI whose complexity per iteration is linear in N , and whose convergence is often subquadratic in N , leading to a relative asymptotic efficiency over naïve linear algebraic techniques.

3 BACKGROUND

3.1 MARKOV RANDOM FIELDS

A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with vertex set \mathcal{V} and edge set \mathcal{E} linking pairs of vertices, can be used to represent the conditional independence structure of a joint distribution $p_{\mathbf{x}}(\cdot)$ over a collection $\mathbf{x} = (x_1, \dots, x_N)$ of N random variables (or disjoint random subvectors). This paper considers the class of distributions represented by undirected graphs, also known as Markov random fields (MRFs).

The topology of an MRF can be characterized, in part, by its set of paths. A path is a sequence of distinct adjacent vertices (v_1, \dots, v_m) where $\{v_k, v_{k+1}\} \in \mathcal{E}, k = 1, \dots, m-1$. If for any two distinct vertices $s, t \in \mathcal{V}$ there is more than one path joining s to t , then \mathcal{G} contains a cycle. A graph without cycles is called a *tree* (or, if it is disconnected, a *forest*).

An MRF can represent conditional independences of the form given by the global Markov condition: For disjoint subsets $A, B, C \subset \mathcal{V}$, $\mathbf{x}_A \perp\!\!\!\perp \mathbf{x}_B \mid \mathbf{x}_C$ iff A and B are graph-separated by C (all paths between a vertex in A and a vertex in B must pass through C). The edge set \mathcal{E} satisfies the pairwise Markov property: For all $i, j \in \mathcal{V}$, $\{i, j\} \notin \mathcal{E}$ iff $\mathbf{x}_i \perp\!\!\!\perp \mathbf{x}_j \mid \mathbf{x}_{\mathcal{V} \setminus \{i, j\}}$. A distribution is said to be Markov with respect to a graph \mathcal{G} if it satisfies the conditional independences implied by \mathcal{G} .

3.2 INFERENCE ON GAUSSIAN MRFS

A multivariate Gaussian distribution in the information form $p_{\mathbf{x}}(\mathbf{x}) \propto \exp\{-\frac{1}{2}\mathbf{x}^T J \mathbf{x} + \mathbf{h}^T \mathbf{x}\}$, with (symmetric, positive definite) *precision* or *inverse covariance matrix* J and potential vector \mathbf{h} , is Markov with respect to a Gaussian MRF (GMRF; Speed and Kiiveri, 1986) if J satisfies the sparsity pattern of \mathcal{E} : $(J)_{i,j} = (J)_{j,i}^T \neq 0 \Leftrightarrow \{i, j\} \in \mathcal{E}$. The parameters of the information form are related to the covariance $P = J^{-1}$ and mean $J^{-1}\mathbf{h}$. Thus, estimating the mean of a Gaussian is equivalent to solving the system of equations

$$J\hat{\mathbf{x}} = \mathbf{h}. \quad (2)$$

Assume without loss of generality¹ that each component x_i of \mathbf{x} is a subvector of dimension $d \in \mathbb{N}^+$, whereby $J \in \mathbb{R}^{Nd \times Nd}$ can be partitioned into an $N \times N$ grid of $d \times d$ block submatrices. Solving (2) by inverting J requires $\mathcal{O}((Nd)^3)$ operations, which can be prohibitively expensive for large N . If the graph contains no cycles, then Gaussian belief propagation (GaBP) (Pearl, 1988; Weiss and Freeman, 2001) can be used to compute the conditional mean, as well as marginal variances, in $\mathcal{O}(Nd^3)$, providing a significant computational savings for large N . For graphs

¹Extension to the case of varying subvector dimensions with $d \triangleq \max_{i \in \mathcal{V}} \dim(\mathbf{x}_i)$ is straightforward.

with cycles, various estimation procedures have been recently developed to exploit available sparsity in the graph (cf. Sections 3.4 and 3.5).

Marginalization and conditioning can be conceptualized as selecting submatrices of P and J , respectively. Let disjoint sets A and B form a partition of $\mathcal{V} = \{1, \dots, N\}$. The marginal distribution $p_{\mathbf{x}_A}(\cdot)$ over \mathbf{x}_A is parameterized by covariance matrix $(P)_{A,A}$, the block submatrix of P corresponding to the rows and columns indexed by A . Similarly, the conditional distribution $p_{\mathbf{x}_A|\mathbf{x}_B}(\cdot|\mathbf{x}_B)$ of \mathbf{x}_A conditioned on $\mathbf{x}_B := \mathbf{x}_B$ is parameterized by the $(J)_{A,A}$ block submatrix of J . In the inferential setting, one has access to J and not P .

3.3 MUTUAL INFORMATION

Mutual information (MI) is an information-theoretic measure of dependence between two (sets of) random variables. Its interpretation as a measure of entropy reduction appeals to its use in uncertainty mitigation (Caselton and Zidek, 1984). For disjoint subsets $A, B, C \subset \mathcal{V}$ (with C possibly empty), conditional mutual information is defined as

$$I(\mathbf{x}_A; \mathbf{x}_B | \mathbf{x}_C) \triangleq h(\mathbf{x}_A | \mathbf{x}_C) - h(\mathbf{x}_A | \mathbf{x}_B, \mathbf{x}_C), \quad (3)$$

where, for continuous random variables \mathbf{x}_V , $h(\cdot)$ is the differential entropy functional

$$h(q_{\mathbf{x}}(\cdot)) = - \int_{\mathcal{X}} q_{\mathbf{x}}(x) \log q_{\mathbf{x}}(x) dx.$$

Note that MI is always nonnegative and is symmetric with respect to its first two arguments. For convenience, we will often use only the index sets (and not the random variables they index) as the arguments of mutual information.

Let $P_{A|C}$ denote the (marginal) covariance of \mathbf{x}_A given \mathbf{x}_C . For multivariate Gaussians, the conditional MI (Cover and Thomas, 2006) is

$$I(A; B | C) = \frac{1}{2} \log \frac{\det(P_{A|C}) \det(P_{B|C})}{\det(P_{A \cup B | C})}. \quad (4)$$

Computing the marginal covariance matrices needed in (4) via matrix inversion (or taking Schur complements) of $J_{A \cup B | C}$ generally requires $\mathcal{O}((Nd)^3)$ operations, even if one is computing *pairwise* MI (i.e., $|A| = |B| = 1$). For Gaussian trees, an efficient algorithm exist for reducing pairwise MI complexity to $\mathcal{O}(Nd^3)$, i.e., linear in the number of vertices (Levine and How, 2013). The main objective of this paper is providing a similar reduction in complexity for loopy Gaussian graphical models.

3.4 EMBEDDED TREES

The embedded trees (ET) algorithm was introduced in (Sudderth, 2002; Wainwright et al., 2000) to iteratively

compute both conditional means and marginal error variances in Gaussian graphical models with cycles. Although the algorithm requires only the identification of subgraphs on which inference is tractable, and extensions to, for example, embedded polygons (Delouille et al., 2006) and embedded hypergraphs (Chandrasekaran et al., 2008) have been considered, this paper will focus for clarity of discussion on embedded trees.

Let $\mathbf{x} \sim \mathcal{N}^{-1}(\mathbf{0}, J)$ be a Gaussian distributed random vector Markov to an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ that contains cycles. Consider an alternatively distributed random vector $\mathbf{x}_{\mathcal{T}} \sim \mathcal{N}^{-1}(\mathbf{0}, J_{\mathcal{T}})$ that is of the same dimension as \mathbf{x} but is instead Markov to a cycle-free subgraph $\mathcal{G}_{\mathcal{T}} = (\mathcal{V}, \mathcal{E}_{\mathcal{T}})$ of \mathcal{G} (in the sense that $\mathcal{E}_{\mathcal{T}} \subset \mathcal{E}$). The tree-shaped (and symmetric, positive definite) inverse covariance matrix $J_{\mathcal{T}}$ can be decomposed as $J_{\mathcal{T}} = J + K_{\mathcal{T}}$, where $K_{\mathcal{T}}$ is any symmetric *cutting matrix* that enforces the sparsity pattern of $J_{\mathcal{T}}$ by zeroing off-diagonal elements of J corresponding to cut edges $\mathcal{E} \setminus \mathcal{E}_{\mathcal{T}}$. Since many cutting matrices $K_{\mathcal{T}}$ will result in a tree-shaped inverse covariance $J_{\mathcal{T}}$ Markov to $\mathcal{G}_{\mathcal{T}}$, attention will be restricted to so-called *regular* cutting matrices, whose nonzero elements are constrained to lie at the intersection of the rows and columns corresponding to the vertices incident to cut edges. Note that $K_{\mathcal{T}}$ can always be chosen such that $\text{rank}(K_{\mathcal{T}})$ is at most $\mathcal{O}(Ed)$, where $E \triangleq |\mathcal{E} \setminus \mathcal{E}_{\mathcal{T}}|$ will be used to denote the number of cut edges.

3.4.1 Conditional Means

Given an initial solution $\hat{\mathbf{x}}^{(0)}$ to (2), the single-tree Richardson iteration (Young, 1971) induced by embedded tree $\mathcal{G}_{\mathcal{T}}$ with cutting matrix $K_{\mathcal{T}}$ and associated inverse covariance $J_{\mathcal{T}} = J + K_{\mathcal{T}}$ is

$$\hat{\mathbf{x}}^{(n)} = J_{\mathcal{T}}^{-1} \left(K_{\mathcal{T}} \hat{\mathbf{x}}^{(n-1)} + \mathbf{h} \right). \quad (5)$$

Thus, each update $\hat{\mathbf{x}}^{(n)}$ is the solution of a synthetic inference problem (2) with precision matrix $\tilde{J} = J_{\mathcal{T}}$ and potential vector $\tilde{\mathbf{h}} = K_{\mathcal{T}} \hat{\mathbf{x}}^{(n-1)} + \mathbf{h}$. This update requires a total of $\mathcal{O}(Nd^3 + Ed^2)$ operations, where $\mathcal{O}(Nd^3)$ is due to solving $\tilde{J} \hat{\mathbf{x}}^{(n)} = \tilde{\mathbf{h}}$ with a tree-shaped graph, and where $\mathcal{O}(Ed^2)$ with $E = |\mathcal{E} \setminus \mathcal{E}_{\mathcal{T}}|$ is due to forming $\tilde{\mathbf{h}}$. In the case that E is at most $\mathcal{O}(N)$, the overall complexity *per iteration* is $\mathcal{O}(Nd^3)$. Letting $\rho(D) \triangleq \max_{\lambda \in \{\lambda_i(D)\}} |\lambda|$ denote the spectral radius of a square matrix D , the asymptotic convergence rate of the single-tree iteration (5) is

$$\rho(J_{\mathcal{T}}^{-1} K_{\mathcal{T}}) = \rho(I - J_{\mathcal{T}}^{-1} J), \quad (6)$$

with convergence to $\hat{\mathbf{x}}$ guaranteed (regardless of $\hat{\mathbf{x}}^{(0)}$) if and only if $\rho(J_{\mathcal{T}}^{-1} K_{\mathcal{T}}) < 1$. Inherent in (5) and (6) is a tradeoff in the choice of embedded structure between the tractability of solving $J_{\mathcal{T}} \hat{\mathbf{x}}^{(n)} = \tilde{\mathbf{h}}$ and the approximation strength of $J_{\mathcal{T}} \approx J$ for fast convergence.

The ET algorithm (Sudderth et al., 2004) is conceptualized as a nonstationary Richardson iteration with multiple matrix splittings of J . Let $\{G_{\mathcal{T}_n}\}_{n=1}^{\infty}$ be a sequence of embedded trees within \mathcal{G} , and let $\{K_{\mathcal{T}_n}\}_{n=1}^{\infty}$ be a sequence of cutting matrices such that $J_{\mathcal{T}_n} = J + K_{\mathcal{T}_n}$ is Markov to $\mathcal{G}_{\mathcal{T}_n}$ for $n = 1, \dots, \infty$. The nonstationary Richardson update is then

$$\hat{\mathbf{x}}^{(n)} = J_{\mathcal{T}_n}^{-1} \left(K_{\mathcal{T}_n} \hat{\mathbf{x}}^{(n-1)} + \mathbf{h} \right), \quad (7)$$

with error $e^{(n)} \triangleq \hat{\mathbf{x}}^{(n)} - \hat{\mathbf{x}}$ that evolves according to

$$e^{(n)} = J_{\mathcal{T}_n}^{-1} K_{\mathcal{T}_n} e^{(n-1)}. \quad (8)$$

The criterion for convergence is when the normalized residual error $\|K_{\mathcal{T}_n}(\hat{\mathbf{x}}^{(n)} - \hat{\mathbf{x}}^{(n-1)})\|_2 / \|\mathbf{h}\|_2$ falls below a specified tolerance $\epsilon > 0$. The sparsity of $K_{\mathcal{T}_n}$ permits the efficient computation of this residual.

When $\{G_{\mathcal{T}_n}, K_{\mathcal{T}_n}\}_{n=1}^{\infty}$ is periodic in n , a convergence rate analysis similar to (6) is given in (Sudderth et al., 2004). It is also demonstrated that using multiple embedded trees can significantly improve the convergence rate. Online adaptive selection of the embedded tree was explored in (Chandrasekaran et al., 2008) by scoring edges according to single-edge walk-sums and forming a maximum weight spanning tree in $\mathcal{O}(|\mathcal{E}| \log |N|)$.

3.4.2 Marginal Variances

Given that $\text{rank}(K_{\mathcal{T}}) \leq 2Ed$ (Sudderth, 2002), where $E = |\mathcal{E} \setminus \mathcal{E}_{\mathcal{T}}|$, an additive rank-one decomposition

$$K_{\mathcal{T}} = \sum_i w_i u_i u_i^T, \quad u_i \in \mathbb{R}^{Nd} \quad (9)$$

can be substituted in the fixed-point equation (Sudderth et al., 2004)

$$P = J_{\mathcal{T}}^{-1} + J_{\mathcal{T}}^{-1} K_{\mathcal{T}} P, \quad (10)$$

yielding

$$P = J_{\mathcal{T}}^{-1} + \sum_i w_i (J_{\mathcal{T}}^{-1} u_i) (P u_i)^T. \quad (11)$$

Solving for the vertex-marginal covariances $P_i = (P)_{i,i}$, $i \in \mathcal{V}$, which are the block-diagonal entries of P , requires:

- solving for the block-diagonal entries of $J_{\mathcal{T}}^{-1}$, with one-time complexity $\mathcal{O}(Nd^3)$ via GaBP;
- solving the synthetic inference problems $J_{\mathcal{T}} z_i = u_i$, for all $\mathcal{O}(Ed)$ vectors u_i of $K_{\mathcal{T}}$ in (9), with one-time total complexity $\mathcal{O}(Nd^3 \cdot Ed) = \mathcal{O}(NEd^4)$ via GaBP;
- solving the synthetic inference problems $J z_i = u_i$, for all $\mathcal{O}(Ed)$ vectors u_i of $K_{\mathcal{T}}$ in (9), with *per iteration* total complexity of $\mathcal{O}(NEd^4)$ operations via ET conditional means (7);

- and assembling the above components via (11).

Note that there exists a decomposition, alternative to (9), of $K_{\mathcal{T}}$ into $\mathcal{O}(Wd)$ rank-one matrices using a cardinality- W vertex cover of $\mathcal{E} \setminus \mathcal{E}_{\mathcal{T}}$ (where $W \leq E$ for any minimal vertex cover); this alternative decomposition requires solving a symmetric quadratic eigenvalue problem (Sudderth et al., 2004).

3.5 COMPETING METHODS

Other methodologies have been proposed to perform inference in loopy graphs. Loopy belief propagation (LBP) is simply parallel belief propagation performed on graphs with cycles; if it converges, it does so to the correct mean but, in general, to incorrect variances (Weiss and Freeman, 2001). Extended message passing augments the original BP messages and provides for convergence to the correct variances, but its complexity is $\mathcal{O}(NL^2)$ in the scalar case, where L is the number of vertices incident to any cut edge, and it requires the full message schedule to be executed to produce an estimate (Plarre and Kumar, 2004). Linear response algorithms can be used to compute pairwise marginal distributions for nonadjacent pairs of vertices, but at a complexity of $\mathcal{O}(N|\mathcal{E}|d^3)$, which may be excessive for large N and $|\mathcal{E}| = \mathcal{O}(N)$ given that conditional MI requires only very specific pairwise marginals (Welling and Teh, 2004).

It is also possible to perform efficient inference if it is known that removal of a subset of \mathcal{V} , called a feedback vertex set (FVS), will induce a tree-shaped subgraph (Liu et al., 2012). The resulting belief propagation-like inference algorithm, called feedback message passing, has complexity $\mathcal{O}(Nk^2)$ for scalar networks, where k is the size of the FVS. If the topological structure of the graphical model is well known *a priori*, or if the graph is learned by an algorithm oriented towards forming FVSs (Liu and Willsky, 2013), then identification of an FVS is straightforward. However, if the graphical model is provided without such identification, it may be computationally expensive to form an FVS of reasonable size, whereas finding a spanning tree (as the ET algorithm does) is comparatively simple.

Various graph sparsification methods have been pursued to find useful substructures that can precondition linear systems of equations (e.g., support graph theory (Bern et al., 2006)). Notably, Spielman and Teng (2011) present a spectral sparsification method for the graph Laplacian (which has scalar edge weights) that permits the solution of diagonally dominant linear systems in near-linear time. In contrast, this paper analyzes the ET sparsifier, which operates on edges with potentially *vectoral* weights and does *not* assume diagonal dominance (which would, for example, guarantee the convergence of LBP (Weiss and Freeman, 2001)).

4 ET MUTUAL INFORMATION QUANTIFICATION (ET-MIQ)

This section describes the application of embedded trees to efficient iterative computation of nonlocal mutual information measures on loopy Gaussian graphs.

It is typically intractable to enumerate all possible selection sets $\mathcal{A} \in 2^{\mathcal{V}}$ and evaluate the resulting MI objective $I(\mathcal{R}; \mathcal{A})$. Often, one balances tractability with performance by using suboptimal selection heuristics with either a priori or online-computable performance bounds (Krause and Guestrin, 2005; Levine and How, 2013). Starting from an empty selection $\mathcal{A} \leftarrow \emptyset$, the greedy heuristic

$$\begin{aligned} a &\leftarrow \operatorname{argmax}_{\{y \in \mathcal{S} \setminus \mathcal{A} : c(y) \leq \beta - c(\mathcal{A})\}} I(\mathcal{R}; y | \mathcal{A}) \\ \mathcal{A} &\leftarrow \mathcal{A} \cup \{a\} \end{aligned} \quad (12)$$

selects one unselected observable variable with the highest marginal increase in objective and continues to do so until the budget is expended. By comparison to (4), the MI evaluations needed to perform a greedy update are of the form

$$I(\mathcal{R}; y | \mathcal{A}) = \frac{1}{2} \log \frac{\det(P_{\mathcal{R}|\mathcal{A}}) \det(P_{\{y\}|\mathcal{A}})}{\det(P_{\mathcal{R} \cup \{y\}|\mathcal{A}})} \quad (13)$$

While inverse covariance matrices obey specific sparsity patterns, covariance matrices are generally dense. Thus two of the determinants in (13) require $\mathcal{O}(|\mathcal{R}|^3 d^3)$ operations to compute. If $|\mathcal{R}|$ is $\mathcal{O}(N)$ (e.g., the graph represents a regular pattern, a constant fraction of which is to be inferred), then such determinants would be intractable for large N . One instead fixes some ordering R over the elements of \mathcal{R} , denoting by r_k its k th element, $R_k = \cup_{i=1}^k \{r_i\}$ its first k elements, and appeal to the chain rule of mutual information:

$$\begin{aligned} I(\mathcal{R}; y | \mathcal{A}) &= I(r_1; y | \mathcal{A}) + I(r_2; y | \mathcal{A} \cup R_1) + \dots \\ &\quad + I(r_{|\mathcal{R}|}; y | \mathcal{A} \cup R_{|\mathcal{R}|-1}). \end{aligned} \quad (14)$$

The advantage of this expansion is twofold. Each term in the summation is a pairwise mutual information term. Given an efficient method for computing marginal covariance matrices (the focus of the remainder of this section), the determinants in (4) can be evaluated in $\mathcal{O}(d^3)$ operations. More pressingly, conditioning in an undirected graphical model removes paths from the graph (by the global Markov property), potentially simplifying the structure over which one must perform the quantification. Therefore, the chain rule converts the problem of evaluating a set mutual information measure $I(\mathcal{R}; y | \mathcal{A})$ into $|\mathcal{R}|$ separate pairwise MI computations that *decrease* in difficulty as the conditioning set expands.

It suffices to describe how to compute *one* of the $|\mathcal{R}|$ terms in the summation (14); the template will be repeated for the

other $|\mathcal{R}| - 1$ terms, but with a modified conditioning set. In the remainder of this section, it is shown how to efficiently compute $I(r; y | C)$ for all $y \in \mathcal{S} \setminus C$ provided some $r \in \mathcal{R}$ and conditioning set $C \subset \mathcal{V} \setminus \{r\}$. Since conditioning on C can be performed by selecting the appropriate submatrix of J corresponding to $\mathcal{V} \setminus C$, it is assumed for clarity of presentation and without loss of generality² that either $C = \emptyset$ or that one is always working with a J resulting from a larger J' that has been conditioned on C . The resulting MI terms, in further simplification of (13), are of the form

$$I(r; y) = \frac{1}{2} \log \frac{\det(P_{\{r\}}) \det(P_{\{y\}})}{\det(P_{\{r,y\}})}, \quad (15)$$

where $P_{\{r\}} = (P)_{r,r}$ and $P_{\{y\}} = (P)_{y,y}$ are the $d \times d$ marginal covariances on the diagonal, and where $P_{\{r,y\}}$ is the $2d \times 2d$ block submatrix of the (symmetric) covariance $P = J^{-1}$:

$$P_{\{r,y\}} = \begin{bmatrix} (P)_{r,r} & (P)_{r,y} \\ (P)_{y,r} & (P)_{y,y} \end{bmatrix}.$$

In addition to the marginal covariances on the diagonal, the $d \times d$ off-diagonal cross-covariance term $(P)_{r,y} = (P)_{y,r}^T$ is needed to complete $P_{\{r,y\}}$. If it were possible to efficiently estimate the d columns of P corresponding to r , all such cross-covariance terms $(P)_{r,y}, \forall y \in \mathcal{S}$, would be available. Therefore, let P be partitioned into columns $\{p_i\}_{i=1}^{Nd}$ and assume without loss of generality that r corresponds to p_1, \dots, p_d . Let e_i be the i th Nd -dimensional axis vector (with a 1 in the i th position). Then $p_i \equiv P e_i, i = 1, \dots, d$, can be estimated using the synthetic inference problem

$$J p_i = e_i. \quad (16)$$

Thus, by comparison to (2) and (7), the first d columns of P can be estimated with a complexity of $\mathcal{O}(Nd^4)$ per ET iteration.

Using the results of Section 3.4.2, the marginal variances can be estimated in $\mathcal{O}(NEd^4)$ per iteration, where $E = |\mathcal{E} \setminus \mathcal{E}_{\mathcal{T}_n}|$ is the number of cut edges. One can subsequently form each matrix $P_{\{r,y\}}, y \in \mathcal{S}$ in $\mathcal{O}(d^2)$ and take its determinant in $\mathcal{O}(d^3)$. Since $|\mathcal{S}| < N$, the ET-MIQ procedure outlined in the section can be used to iteratively estimate the set $\{I(r; y)\}_{y \in \mathcal{S}}$ with total complexity $\mathcal{O}(NEd^4)$ operations per iteration. Returning to the greedy selection of (12) and the chain rule of (14), given a subset $\mathcal{A} \subset \mathcal{S}$ of previous selections, the set of marginal gains $\{I(\mathcal{R}; y | \mathcal{A})\}_{y \in \mathcal{S} \setminus \mathcal{A}}$ can be estimated in $\mathcal{O}(N|\mathcal{R}|Ed^4)$ operations per iteration.

²Alternatively, the unconditioned J can be used by treating conditioned vertices as blocked (not passing messages) and by zeroing the elements of \mathbf{h} and $\hat{\mathbf{x}}^{(n)}$ in (5) corresponding to C .

5 EXPERIMENTS

5.1 ALTERNATIVE METHODS

In order to demonstrate the comparative performance of the ET-MIQ procedure of Section 4, alternative methods for computing mutual information in Gaussian graphs – two based on matrix inversion, and one based exclusively on estimating columns of P – are briefly described.

5.1.1 Naïve Inversion

Whenever a mutual information term of the form $I(A; B|C)$ is needed, the `NaïveInversion` procedure conditions J on C and computes the marginal covariance matrices $P_{A \cup B|C}$, $P_{A|C}$, and $P_{B|C}$ of (4) using standard matrix inversion, which is $\mathcal{O}(N^3 d^3)$. A greedy selection update, which requires computing marginal information gain scores $\{I(\mathcal{R}; y|\mathcal{A})\}_{y \in \mathcal{S} \setminus \mathcal{A}}$, thereby requires $\mathcal{O}(N^3 |\mathcal{S}| d^3)$ operations using this procedure.

5.1.2 Block Inversion

Intuitively, the `NaïveInversion` procedure appears wasteful even for an inversion-based method, as it repeats many of the marginalization operations needed to form $\{I(\mathcal{R}; y|\mathcal{A})\}_{y \in \mathcal{S} \setminus \mathcal{A}}$. The `BlockInversion` procedure attempts to rectify this. Given a previous selection set \mathcal{A} , `BlockInversion` conditions J on \mathcal{A} and marginalizes out nuisances $\mathcal{U} \setminus \mathcal{R}$ (along with infeasible observation selections $\{y \in \mathcal{S} \setminus \mathcal{A} \mid c(y) > \beta - c(\mathcal{A})\}$) using Schur complements. The complexity of this approach, for each greedy update, is $\mathcal{O}(|\mathcal{S}|^4 + |\mathcal{R}||\mathcal{S}|^3 + |\mathcal{R}|^3|\mathcal{S}| + N^3)$. `BlockInversion` has the same worst-case asymptotic complexity of $\mathcal{O}(N^3 |\mathcal{S}| d^3)$ as `NaïveInversion` but may achieve a significant reduction in computation depending on how $|\mathcal{R}|$ and $|\mathcal{S}|$ scale with N .

5.1.3 ColumnET

The `ColumnET` procedure uses nonstationary embedded tree estimation of specific columns of P to compute all information measures. That is to say, no marginal error variance terms are computed (cf. Section 3.4.2). Given a previous selection set \mathcal{A} , and an ordering R over \mathcal{R} , the columns of $P_{\cdot|\mathcal{A} \cup R_{k-1}}$ corresponding to $\{r_k\} \cup \mathcal{S} \setminus \mathcal{A}$ are estimated via (7) and (16). The complexity of a greedy update using `ColumnET` is $\mathcal{O}(N|\mathcal{R}||\mathcal{S}|d^4)$ operations per ET iteration.

5.2 “HOOP-TREE” EXAMPLES

To investigate the performance benefits of ET-MIQ, we consider a subclass of scalar ($d = 1$) loopy graphs containing m simple cycles (achordal “hoops”) of length l , where cycles may share vertices but no two cycles may share edges. The structure of this graph resembles a macro-tree

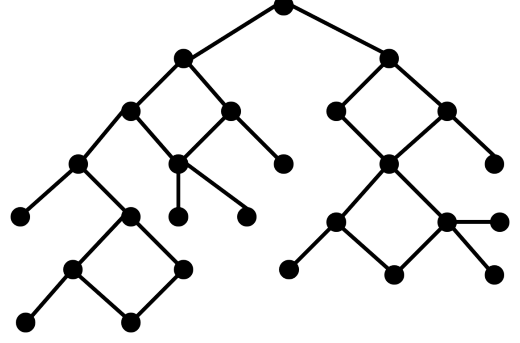


Figure 1: Example of a hoop-tree with 4-vertex cycles.

over hoop subcomponents (a “hoop-tree”; see Figure 1). Any embedded tree on this graph must only cut m edges ($E = m$), one for each l -cycle. This class of graphs is useful for benchmarking purposes, as it permits randomization without requiring the subsequent enumeration of loops via topological analysis, which may be computationally expensive and thus inefficient for testing.

For each problem *instance*, we generate a random hoop-tree $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ of size $|\mathcal{V}| = N$. To generate a corresponding inverse covariance J , we sample $(J)_{i,j} \sim \text{uniform}([-1, 1])$ for each $\{i, j\} \in \mathcal{E}$, and sample $(J)_{i,i} \sim \text{Rayleigh}(1)$, with the diagonal rescaled to enforce the positive definiteness of J . We then randomly label vertices in \mathcal{V} as belonging to \mathcal{S} or \mathcal{U} (or neither), set a budget $\beta \propto |\mathcal{S}|$, and sample an integer-valued additive cost function $c(\cdot)$ such that $c(s) \sim \text{uniform}([1, \gamma\beta])$ for some $\gamma \in [0, 1]$ and all $s \in \mathcal{S}$, and such that $c(\mathcal{A}) = \sum_{a \in \mathcal{A}} c(a)$ for all $\mathcal{A} \subseteq \mathcal{S}$.

Let $\mathcal{G}_{\mathcal{T}_1}$ and $K_{\mathcal{T}_1}$ be the embedded subtree and associated regular cutting matrix formed by cutting the edge of each l -cycle with the highest absolute precision parameter $|(J)_{i,j}|$. Guided by the empirical results of Sudderth et al. (2004), the second embedded tree $\mathcal{G}_{\mathcal{T}_2}$ is selected such that in every l -cycle, $K_{\mathcal{T}_2}$ cuts the edge farthest from the corresponding cut edge in the $\mathcal{G}_{\mathcal{T}_1}$ (modulo some tie-breaking for odd l).

Figure 2 summarizes a comparison of ET-MIQ against `NaïveInversion`, `BlockInversion`, and `ColumnET` in terms of the mean runtime to complete a full greedy selection. Random networks of size N were generated, with $|\mathcal{R}| = 5$ and $|\mathcal{S}| = 0.3N$. The alternative methods were suppressed when they began to take prohibitively long to simulate (e.g., $N = 1200$ for `BlockInversion` and `ColumnET`).

The runtime of ET-MIQ, which vastly outperforms the alternative methods for this problem class, appears to grow superlinearly, but subquadratically, in N (approximately, bounded by $\mathcal{O}(N^{1.7})$). The growth rate is a confluence of three factors: the $\mathcal{O}(N|\mathcal{R}|Ed^4)$ complexity per Richardson iteration of updating $\{I(\mathcal{R}; y|\mathcal{A})\}_{y \in \mathcal{S} \setminus \mathcal{A}}$; the number

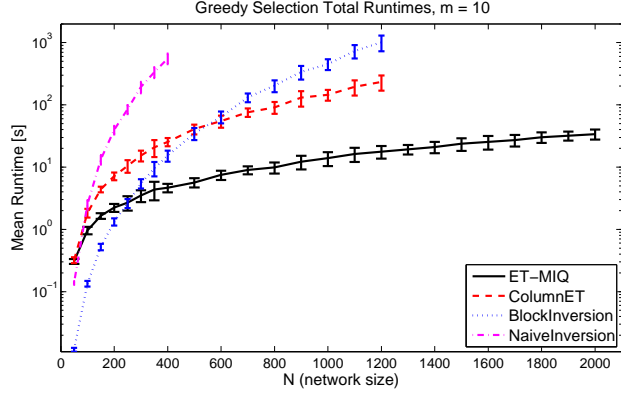


Figure 2: Mean runtime of the full greedy selection as a function of the network size N for randomized loopy graphs with $m = 10$ simple cycles of length $l = 4$.

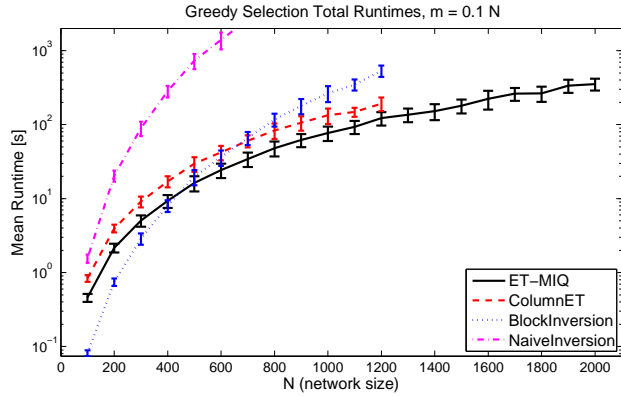


Figure 3: Mean runtime of the full greedy selection as a function of the network size N for randomized loopy graphs with $m = 0.1N$ simple cycles of length $l = 4$. As predicted, in the case where $m = \mathcal{O}(N)$, the ET-based algorithms have the same asymptotic complexity; ET-MIQ has a lower constant factor.

of Richardson iterations until the normalized residual error converges to a fixed tolerance of $\epsilon = 10^{-10}$; and the growth rate of $|\mathcal{S}|$ as a function of N , which indirectly affects the runtime through the budget β by permitting larger selection sets, and hence more rounds of greedy selection. To better disambiguate the second and third factors, we studied how the number of Richardson iterations to convergence (for a random input \mathbf{h} ; cf. (2)) varies as a function of N and found no significant correlation in the case where m is constant (not a function of N). The median iteration count was 7, with standard deviation of 0.6 and range 5-9 iterations.

We also considered the effect of letting m , the number of cycles in the graph, vary with N . A runtime comparison for $m = 0.1N$ is shown in Figure 3. Given that $E = m = \mathcal{O}(N)$ and $|\mathcal{R}| = \mathcal{O}(1)$, ET-MIQ has an asymptotic complexity of $\mathcal{O}(N|\mathcal{R}|Ed^4) = \mathcal{O}(N^2)$. Similarly, the complexity of ColumnET is $\mathcal{O}(N|\mathcal{R}||\mathcal{S}|d^4) = \mathcal{O}(N^2)$. Fig-

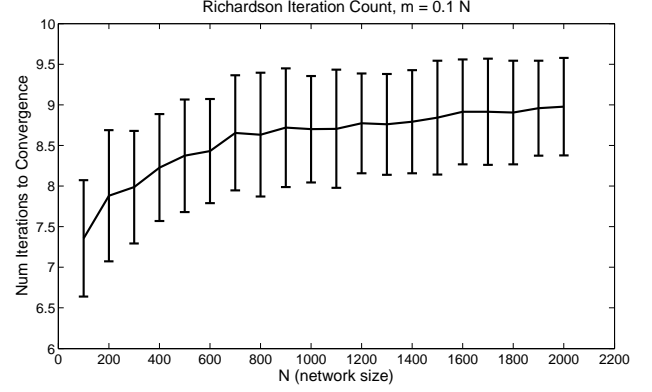


Figure 4: Number of Richardson iterations until convergence, for $m = 0.1N$ cycles.

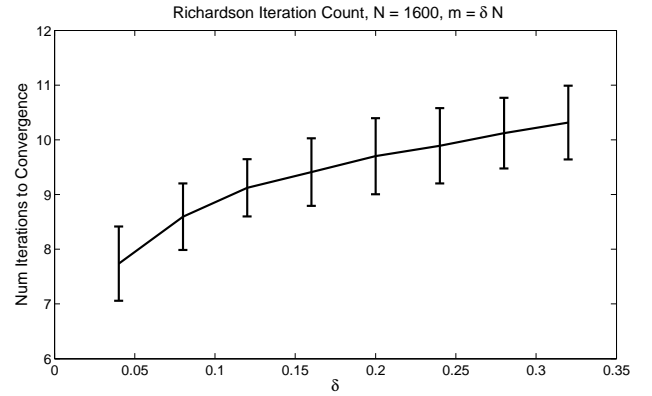


Figure 5: Number of Richardson iterations until convergence, for $N = 1600$ vertices, $m = \delta N$ cycles.

ure 3 confirms this agreement of asymptotic complexity, with ET-MIQ having a lower constant factor.

We repeated the convergence study for $m = 0.1N$ and varying $N \in [100, 2000]$ (see Figure 4). The mean iteration count appears to grow sublinearly in N ; the *actual* increase in iteration count over N is quite modest.

The relationship between the convergence and the problem structure was more clearly illustrated when we fixed a network size of $N = 1600$ and varied the number of 4-vertex cycles $m = \delta N$, for $\delta \in [0.04, 0.32]$ (see Figure 5). The cycle fraction δ is strongly correlated with the iteration count – and even slightly more correlated with its log – suggesting an approximately linear (and perhaps marginally sublinear) relationship with δ , albeit with a very shallow slope.

6 DISCUSSION

This paper has presented a method of computing nonlocal mutual information in Gaussian graphical models containing both cycles and nuisances. The base computations are

iterative and performed using trees embedded in the graph. We assess the proposed algorithm, ET-MIQ, and its alternatives (cf. Sections 3.5 and 5.1) in terms of the asymptotic complexity of performing a greedy update. For ET-MIQ, per-iteration complexity is $\mathcal{O}(N|\mathcal{R}|Ed^4)$, where N is the number of vertices in the network, $\mathcal{R} \subset \mathcal{V}$ is set of relevant latent variables that are of inferential interest, E is the number of edges cut to form the embedded tree, and d is the dimension of each random vector indexed by a vertex of the graph. Let κ denote the expected number of Richardson iterations to convergence of ET-MIQ, which is a direct function of the eigenproperties of the loopy precision matrix and its embedded trees and an indirect function of the other instance-specific parameters (number of cycles, network size, etc.). The experimental results of Section 5 suggest that the proposed algorithm, ET-MIQ, achieves significant reduction in computation over inversion-based methods, which have a total complexity of $\mathcal{O}(N^3|S|d^3)$, where $S \subset \mathcal{V}$ is the set of observable vertices that one has the option of selecting to later realize.

Based on the asymptotic complexities, we expect ET-MIQ would continue to achieve a significant reduction in computation for large networks whenever $|\mathcal{R}|Ed\kappa = o(N^2|S|)$. Typically, the vertex dimension d is not a function of the network size. For dense networks ($|\mathcal{E}| = \mathcal{O}(N^2)$), we would not expect significant performance improvements using ET-MIQ; however, it is often the case that \mathcal{E} is sparse in the sense that the number of cut edges $E = \mathcal{O}(N)$. With $|S| = \mathcal{O}(N)$ (the number of available observations growing linearly in the network size), asymptotic benefits would be apparent for $|\mathcal{R}|\kappa = o(N^2)$. Since we suspect κ grows sublinearly (and very modestly) in N , and whichever system utilizing the graphical model is free to choose \mathcal{R} , we expect that ET-MIQ would be beneficial for efficiently quantifying information in a wide class of active inference problems on Gaussian graphs.

The methods described in this paper are exact in the sense that all mutual information measures are estimated to within a specified tolerance. If the computational cost of quantifying mutual information were constrained (e.g., in a distributed estimation framework with communication costs), it may be of interest to develop algorithms for allowing prioritized approximation depending on how sensitive the overall information reward is to these conditional mutual information terms. In addition to algorithms for adaptively selecting embedded trees to hasten convergence, Chandrasekaran et al. (2008) propose methods for choosing and updating only a subset of variables in each Richardson iteration. If, in an essentially dual problem to (1), the cost of sensor selections were to be minimized subject to a quota constraint on the minimum amount of collected information, the ability to truncate information quantification when a subset of the graph falls below an informativeness threshold would be of potential interest, which we intend

to explore in future work.

Acknowledgements

The authors thank Dr. John W. Fisher III for helpful discussions during the preparation of this paper. This work was supported by DARPA Mathematics of Sensing, Exploitation and Execution (MSEE).

References

- M. Bern, J. R. Gilbert, B. Hendrickson, N. Nguyen, and S. Toledo. Support-graph preconditioners. *SIAM Journal on Matrix Analysis and Applications*, 27(4):930–951, 2006.
- W. F. Caselton and J. V. Zidek. Optimal monitoring network designs. *Statistics and Probability Letters*, 2(4): 223–227, 1984.
- V. Chandrasekaran, J. K. Johnson, and A. S. Willsky. Estimation in Gaussian graphical models using tractable subgraphs: A walk-sum analysis. *IEEE Transactions on Signal Processing*, 56(5):1916–1930, May 2008.
- M. J. Choi, V. Y. F. Tan, A. Anandkumar, and A. S. Willsky. Learning latent tree graphical models. *Journal of Machine Learning Research*, 12:1771–1812, May 2011.
- T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley, second edition, 2006.
- V. Delouille, R. Neelamani, and R. G. Baraniuk. Robust distributed estimation using the embedded subgraphs algorithm. *IEEE Transactions on Signal Processing*, 54: 2998–3010, 2006.
- D. Golovin and A. Krause. Adaptive submodularity: A new approach to active learning and stochastic optimization. In *Proc. Int. Conf. on Learning Theory (COLT)*, 2010.
- C. Ko, J. Lee, and M. Queyranne. An exact algorithm for maximum entropy sampling. *Operations Research*, 43: 684–691, 1995.
- D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.
- A. Kraskov, H. Stögbauer, and P. Grassberger. Estimating mutual information. *Physical Review E*, 69(6):066138, jun 2004.
- A. Krause and C. Guestrin. Near-optimal nonmyopic value of information in graphical models. In *Proc. Uncertainty in Artificial Intelligence (UAI)*, 2005.
- A. Krause and C. Guestrin. Optimal value of information in graphical models. *Journal of Artificial Intelligence Research*, 35:557–591, 2009.
- A. Krause, A. Singh, and C. Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9:235–284, 2008.

- D. Levine and J. P. How. Sensor selection in high-dimensional Gaussian trees with nuisances. In C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 26, pages 2211–2219, 2013.
- Y. Liu and A. S. Willsky. Learning Gaussian graphical models with observed or latent FVSs. In *Advances in Neural Information Processing Systems (NIPS)*, volume 26, 2013.
- Y. Liu, V. Chandrasekaran, A. Anandkumar, and A. S. Willsky. Feedback message passing for inference in Gaussian graphical models. *IEEE Transactions on Signal Processing*, 60(8):4135–4150, Aug 2012.
- G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of approximations for maximizing submodular set functions. *Mathematical Programming*, 14:489–498, 1978.
- J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufman, San Mateo, CA, 1988.
- K. H. Plarre and P. R. Kumar. Extended message passing algorithm for inference in loopy Gaussian graphical models. *Ad Hoc Networks*, 2:153–169, 2004.
- T. P. Speed and H. T. Kiiveri. Gaussian Markov distributions over finite graphs. *Annals of Statistics*, 14(1):138–150, mar 1986.
- D. A. Spielman and S.-H. Teng. Spectral sparsification of graphs. *SIAM Journal on Computing*, 40(4):981–1025, 2011.
- E. B. Sudderth. Embedded trees: Estimation of Gaussian processes on graphs with cycles. Master’s thesis, Massachusetts Institute of Technology, February 2002.
- E. B. Sudderth, M. J. Wainwright, and A. S. Willsky. Embedded trees: Estimation of Gaussian processes on graphs with cycles. *IEEE Transactions on Signal Processing*, 52(11):3136–3150, November 2004.
- M. J. Wainwright, E. B. Sudderth, and A. S. Willsky. Tree-based modeling and estimation of Gaussian processes on graphs with cycles. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems (NIPS)*, volume 13. MIT Press, Nov 2000.
- Y. Weiss and W. T. Freeman. Correctness of belief propagation in Gaussian graphical models of arbitrary topology. *Neural Computation*, 13(10):2173–2200, 2001.
- M. Welling and Y. W. Teh. Linear response algorithms for approximate inference in graphical models. *Neural Computation*, 16(1):197–221, 2004.
- J. L. Williams, J. W. Fisher III, and A. S. Willsky. Performance guarantees for information theoretic active inference. In M. Meila and X. Shen, editors, *Proc. Eleventh Int. Conf. on Artificial Intelligence and Statistics*, pages 616–623, 2007.
- D. M. Young. *Iterative Solution of Large Linear Systems*. Academic, New York, 1971.

CoRE Kernels

Ping Li

Department of Statistics and Biostatistics
Department of Computer Science
Rutgers University
Piscataway, NJ 08854, USA
pingli@stat.rutgers.edu

Abstract

The term “CoRE kernel” stands for *correlation-resemblance kernel*. In many real-world applications (e.g., computer vision), the data are often high-dimensional, sparse, and non-binary. We propose two types of (nonlinear) CoRE kernels for non-binary sparse data and demonstrate the effectiveness of the new kernels through a classification experiment. CoRE kernels are simple with no tuning parameters. However, training nonlinear kernel SVM can be costly in time and memory and may not be always suitable for truly large-scale industrial applications (e.g., search). In order to make the proposed CoRE kernels more practical, we develop basic probabilistic hashing (approximate) algorithms which transform nonlinear kernels into linear kernels.

1 INTRODUCTION

The use of high-dimensional data has become popular in practice, especially in search, natural language processing (NLP), and computer vision. For example, Winner of 2009 PASCAL image classification challenge [27] used 4 million (non-binary) features. [5, 25, 28] mentioned datasets with billions or even trillions of features.

For text data, the use of extremely high-dimensional representations (e.g., n -grams) is the standard practice. In fact, binary representations for text data could be sufficient if the order of n -grams is high enough. On the other hand, in current practice of computer vision, it is still more common to use non-binary feature representations, for example, *local coordinate coding (LCC)* [29, 27]. It is often the case that in practice high-dimensional non-binary features might be appropriately sparsified without hurting the performance of subsequent tasks (e.g., classification). However simply binarizing the features will often incur loss of accuracies, sometimes significantly so. See Table 1 for an illustration of such a phenomenon.

Our contribution in this paper is the proposal of two types of (nonlinear) “CoRE” kernels, where “CoRE” stands for “correlation-resemblance”, for non-binary sparse data. Interestingly, using CoRE kernels leads to improvement in classification accuracies (in some cases significantly so) on a variety of datasets (see Table 2).

For practical large-scale applications, naive implementations of nonlinear kernels may be too costly (in time and/or memory), while linear learning methods (e.g., linear SVM or logistic regression) are extremely popular in industry. The proposed CoRE kernels would be facing the same challenge. To address this critical issue, we also develop basic hashing algorithms which approximate the CoRE kernels by linear kernels. These new hashing algorithms allow us to take advantage of highly efficient (batch or stochastic) linear learning algorithms, e.g., [15, 24, 1, 8].

In the rest of this section, we first review the definitions of correlation and resemblance, then we provide an experimental study to illustrate the loss of classification accuracies when sparse data are binarized.

1.1 Correlation

We assume a data matrix of size $n \times D$, i.e., n observations in D dimensions. Consider, without loss of generality, two data vectors $u, v \in \mathbb{R}^D$. The correlation is simply the normalized inner product defined as follows

$$\rho = \rho(u, v) = \frac{\sum_{i=1}^D u_i v_i}{\sqrt{\sum_{i=1}^D u_i^2 \sum_{i=1}^D v_i^2}} = \frac{A}{\sqrt{m_1 m_2}}, \quad (1)$$

$$\text{where } A = \sum_{i=1}^D u_i v_i, \quad m_1 = \sum_{i=1}^D u_i^2, \quad m_2 = \sum_{i=1}^D v_i^2$$

It is well-known that $\rho(u, v)$ constitutes a positive definite and linear kernel, which is one of the reasons why correlation is very popular in practice.

1.2 Resemblance

For binary data, the resemblance is commonly used:

$$R = R(u, v) = \frac{a}{f_1 + f_2 - a}, \quad (2)$$

$$\text{where } f_1 = \sum_{i=1}^D 1\{u_i \neq 0\}, \quad f_2 = \sum_{i=1}^D 1\{v_i \neq 0\},$$

$$a = \sum_{i=1}^D 1\{u_i \neq 0\} 1\{v_i \neq 0\}$$

It was shown in [22] that the resemblance defines a type of positive definite (nonlinear) kernel. In this study, we will combine correlation and resemblance to define two new types of nonlinear kernels.

1.3 Linear SVM Experiment

Table 1 lists the datasets, which are non-binary and sparse. The table also presents the test classification accuracies using linear SVM on both the original (non-binary) data and the binarized data. The results in the table illustrate the noticeable drop of accuracies by using only binarized data.¹

Available at the UCI repository, *Youtube* is a multi-view dataset, and we choose the largest set of features (audio) for our experiment. *M-Basic*, *M-Rotate*, and *MNIST10k* were used in [18] for testing *abc-logitboost* and *abc-mart* [17] (and comparisons with deep learning [16]). For *RCV1*, we use a subset of the original testing examples (to facilitate efficient kernel computation later needed in the paper).

Table 1: Classification accuracies using linear SVM (LIBLINEAR [8]) on sparse non-binary data. As we always normalize data to unit norm, the correlation kernel ρ is naturally used in our study. We experiment with the l_2 -regularized linear SVM (with a regularization parameter “ C ”) and report the best test accuracies from a wide range of C values. Using binarized data (i.e., the last column), the test accuracies drop very noticeably in most datasets.

Dataset	#Train	#Test	Linear	Lin. Bin.
M-Basic	12,000	50,000	90.0%	88.9%
MNIST10k	10,000	60,000	90.0%	88.8%
M-Rotate	12,000	50,000	48.0%	44.4%
RCV1	20,242	60,000	96.3%	95.6%
USPS	7,291	2,007	91.8%	87.4%
Youtube	11,930	97,934	47.6%	46.5%

Figure 1 provides more detailed classification accuracy results for a wide range of C values, where C is the usual l_2 -regularization parameter in linear SVM.

¹For all datasets except *USPS*, we used “0” as the threshold to binarize the data. For *USPS*, since it contains many very small entries, we used a threshold which is slightly different from zero.

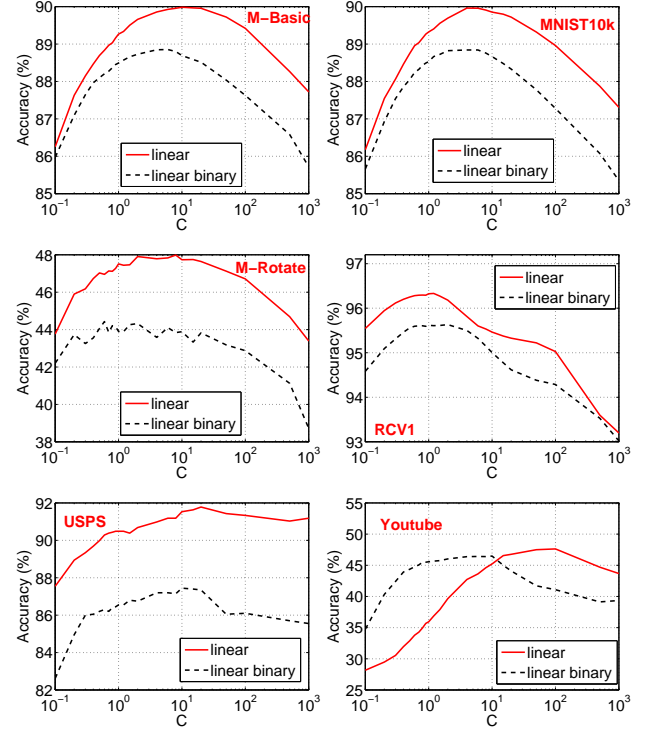


Figure 1: Test classification accuracies for both the original (non-binary, solid) and the binarized (dashed) data, using l_2 -regularized linear SVM with a regularization parameter C . We present results for a wide range of C values. The best (highest) values are summarized in Table 1.

While linear SVM is extremely popular in industrial practice, it is often not as accurate. Our proposed CoRE kernels will be able to produce noticeably more accurate results.

2 CORE KERNELS

We propose two types of CoRE kernels, which combine resemblance with correlation, for sparse non-binary data. Both kernels are positive definite. We will demonstrate the effectiveness of the two CoRE kernels using the same datasets in Table 1 and Figure 1.

2.1 CoRE Kernel, Type 1

The first type of CoRE kernel is basically the product of correlation ρ and the resemblance R , i.e.,

$$K_{C,1} = K_{C,1}(u, v) = \rho R \quad (3)$$

Later in the paper we will express $K_{C,1}$ as an (expectation of) inner product, i.e., $K_{C,1}$ is obviously positive definite.

If the data are fully dense (i.e., no zero entries), then $R = 1$ and $K_{C,1} = \rho$. On the other hand, if the data are binary, then $\rho = \frac{a}{\sqrt{f_1 f_2}}$ and $K_{C,1} = \frac{a}{\sqrt{f_1 f_2} (f_1 + f_2 - a)}$. See (2) for the definitions of f_1, f_2, a .

2.2 CoRE Kernel, Type 2

The second type of CoRE kernel perhaps appears less intuitive than the first type:

$$K_{C,2} = K_{C,2}(u, v) = \rho \frac{\sqrt{f_1 f_2}}{f_1 + f_2 - a} = \frac{\rho R}{a / \sqrt{f_1 f_2}} \quad (4)$$

If the data are binary, then $K_{C,2} = R$. We will, later in the paper, also write $K_{C,2}$ as an expectation of inner product to confirm it is also positive definite.

2.3 Kernel SVM Experiment

Figure 2 presents the classification accuracies on the same 6 datasets as in Figure 1 and Table 1, using nonlinear kernel SVM with three different kinds of kernels: CoRE Type 1, CoRE Type 2, and resemblance. We can see that resemblance (which only uses binary information of the data) does not perform as well as CoRE kernels.

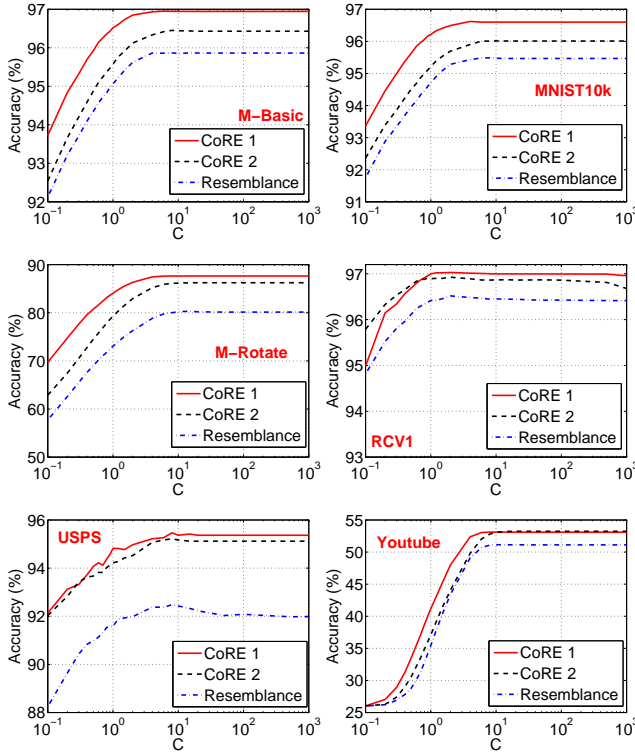


Figure 2: Test classification accuracies using nonlinear kernel SVM and three types of kernels: CoRE Type 1, CoRE Type 2, and resemblance. We use the LIBSVM pre-computed kernel functionality. Compared with the results of linear SVM in Figure 1, we can see CoRE kernels and resemblance kernel perform better (or much better, especially on *M-Rotate* dataset). The best results (highest points on the curves) are summarized in Table 2.

The best results in Figure 2 are summarized in Table 2. It is interesting to compare them with the test accuracies of linear SVM in Table 1 and Figure 1. We can see that CoRE kernels perform very well, without using additional

tuning parameters. In fact, if we compare the best results in [16, 18] (e.g., RBF SVM, abc-boosting, or deep learning) on MNIST10k, M-Rotate, and M-Basic, we will see that CoRE kernels (with no tuning parameters) can achieve the same (or similar) performance.

Table 2: Best test classification accuracies (in %) for five different kernels. The first two columns (i.e., “linear” and “linear binary”) are already shown in Table 1.

Dataset	Lin.	Lin. Bin.	Res.	CoRE1	CoRE2
M-Basic	90.0	88.9	95.9	97.0	96.5
MNIST10k	90.0	88.8	95.5	96.6	96.0
M-Rotate	48.0	44.4	80.3	87.6	86.2
RCV1	96.3	95.6	96.5	97.0	96.9
USPS	91.8	87.4	92.5	95.5	95.2
Youtube	47.6	46.5	51.1	53.1	53.2

We shall mention that our experiments can be fairly easily reproduced because all datasets are public and we use standard SVM packages (LIBSVM and LIBLINEAR) without any modifications. We also provide the results for a wide range of C values in Figure 1 and Figure 2. Note that, because we use pre-computed kernel functionality of LIBSVM (which consumes very substantial memory to store the kernel matrix), we only experiment with training data of moderate sizes, to ensure repeatability (by other researchers without access to machines with large memory).²

2.4 Challenges with Nonlinear Kernel SVM

[2, Section 1.4.3] mentioned three main computational issues of kernels summarized as follows:

1. *Computing kernels is very expensive.*
2. *Computing a full kernel matrix is wasteful, because not all pairwise kernel values are used during training.*
3. *The kernel matrix does not fit in memory.* The cost of storing the full kernel matrix in the memory is $O(n^2)$, which is not realistic for most PCs even for merely 10^5 , while the industry has used training data with billions of examples. Thus, kernel evaluations are often conducted *on the fly*, which means the computational cost is dominated by kernel evaluations.

In fact, evaluating kernels on-demand would encounter another serious (and often common) issue if the dataset itself is too big for the memory.

All these crucial issues motivate us to develop hashing algorithms to approximate CoRE kernels by linear kernels.

²At the time this paper was written, the implementation of LIBSVM restricted the maximum size of the kernel matrix. The LIBSVM team recently has made effort on this issue and it is expected such a restriction will be removed in the new release. We highly appreciate Dr. Chih-Jen Lin and his team for the efforts.

2.5 Benefits of Hashing

Our goal is to develop good probabilistic hashing algorithms to (approximately) transform our proposed nonlinear CoRE kernels into linear kernels. Once we have the new data representations (i.e., the hashed data), we can use highly efficient batch or stochastic linear methods for training SVM (or logistic regression) [15, 24, 1, 8].

Another benefit of hashing would be in the context of approximate near neighbor search because probabilistic hashing provides a (often good) strategy for space partitioning (i.e., bucketing) which will help reduce the search time (i.e., no need to scan all data points). Our proposed hashing methods can be modified to become an instance of *locality sensitive hashing (LSH)* [13] in the space of CoRE kernels.

At this stage, we will focus on developing hashing algorithms for CoRE kernels based on the standard *random projection* and *minwise hashing* methods. There will be plenty of room for improvement which we leave for future work.

We first provide a review of the two basic building blocks.

3 REVIEW OF RANDOM PROJECTIONS AND MINWISE HASHING

Typically, the method of random projections is used for dense high-dimensional data, while the method of minwise hashing is very useful for sparse (often binary) data. The proposed hashing algorithms for CoRE kernels combine random projections and minwise hashing.

3.1 Random Projections

Consider two vectors $u, v \in \mathbb{R}^D$. The idea of random projection is simple. We first generate a random vector of i.i.d. entries $r_i, i = 1$ to D , and then compute the inner products as the hashed values:

$$P(u) = \sum_{i=1}^D u_i r_i, \quad P(v) = \sum_{i=1}^D v_i r_i \quad (5)$$

For the convenience of theoretical analysis, we adopt the choice of $r_i \sim N(0, 1)$, which is a typical choice in the literature. Several variants of random projections like [21, 28] are essentially equivalent, as analyzed in [22].

In this study, we always assume the data are normalized, i.e., $\sum_{i=1}^D u_i^2 = \sum_{i=1}^D v_i^2 = 1$. Note that computing the l_2 norms of all the data points only requires scanning the data once which is anyway needed during data collection/processing. For normalized data, it is known that $E[P(u)P(v)] = \rho$. In order to estimate ρ , we need to use k random projections to generate $P_j(u), P_j(v), j = 1$ to k , and estimate ρ by $\frac{1}{k} \sum_{j=1}^k P_j(u)P_j(v)$, which is also an inner product. This means we can directly use the projected data to build a linear classifier.

3.2 Minwise Hashing

The method of minwise hashing [3] is very popular for computing set similarities, especially for industrial applications, for example, [3, 9, 12, 26, 14, 7, 11, 23, 4].

Consider the space of the column numbers: $\Omega = \{1, 2, 3, \dots, D\}$. We assume a random permutation $\pi : \Omega \rightarrow \Omega$ and apply π on the coordinates of both vectors u and v . For example, consider $D = 4, u = [0, 0.45, 0.89, 0]$ and $\pi : 1 \rightarrow 3, 2 \rightarrow 1, 3 \rightarrow 4, 4 \rightarrow 2$. Then the permuted vector becomes $\pi(u) = [0.45, 0, 0, 0.89]$. In this example, the first nonzero column of $\pi(u)$ is 1, and the corresponding value of the coordinate is 0.45. For convenience, we introduce the following notation:

$$L(u) = \text{location of first nonzero entry of } \pi(u) \quad (6)$$

$$V(u) = \text{value of first nonzero entry of } \pi(u) \quad (7)$$

In this example, we have $L(u) = 1$ and $V(u) = 0.45$.

The well-known *collision probability*

$$\Pr(L(u) = L(v)) = R(u, v) = R \quad (8)$$

can be used to estimate the resemblance R . To do so, we need to generate k permutations $\pi_j, j = 1$ to k .

4 HASHING CORE KERNELS

The goal is to develop unbiased linear estimators of CoRE Kernels $K_{C,1}$ and $K_{C,2}$. Linear estimators can be written as inner products. We assume that we have already conducted random projections and minwise hashing k times. In other words, for each data vector u , we have the hashed values $P_j(u), L_j(u), V_j(u), j = 1$ to k . Recall the definitions of P_j, L_j, V_j in (5), (6), and (7), respectively.

4.1 Hashing Type 1 CoRE Kernel

Our proposed estimator of $K_{C,1}$ is

$$\hat{K}_{C,1}(u, v) = \sum_{j=1}^k P_j(u)P_j(v)1\{L_j(u) = L_j(v)\} \quad (9)$$

The following Theorem 1 shows $\hat{K}_{C,1}$ is an unbiased estimator and provides its variance.

Theorem 1

$$E(\hat{K}_{C,1}) = K_{C,1} \quad (10)$$

$$\text{Var}(\hat{K}_{C,1}) = \frac{1}{k} \{(1 + 2\rho^2)R - \rho^2 R^2\} \quad (11)$$

Proof: See Appendix A. \square

A simple argument can show that $\hat{K}_{C,1}$ could be written as an inner product and hence $K_{C,1}$ is positive definite. Although this fact is obvious since $K_{C,1}$ is a product of two positive definite kernels, we would like to present a constructive proof because the construction is basically the same procedure for expanding the hashed data before feeding them to a linear SVM solver.

Recall, L_j is the location of the first nonzero after minwise hashing. Basically, we can view $L_j(u)$ equivalently as a vector of length D whose coordinates are all zero except the $L_j(u)$ -th coordinate. The value of the only nonzero coordinate will be $P_j(u)$. For example, suppose $D = 4$, $L_j(u) = 2$, $P_j(u) = 0.1$. Then the equivalent vector would be $[0, 0.1, 0, 0]$. With k projections and k permutations, we can have k such vectors. This way, we can write $\hat{K}_{C,1}$ as an inner product of two $D \times k$ -dimensional sparse vectors.

Note that the input data format of standard SVM packages is the sparse format. For linear SVM, the cost is essentially determined by the number of nonzeros (in this case, k), not much to do with the dimensionality (unless it is too high). If D is too high, then we can adopt the standard trick of *b-bit minwise hashing* [22] by only using the lowest b bits of $L_j(u)$. This will lead to an efficient implementation.

4.2 Hashing Type 2 CoRE Kernel

Our proposed estimator for Type 2 CoRE Kernel is

$$\hat{K}_{C,2} = \frac{\sqrt{f_1 f_2}}{k} \sum_{j=1}^k V_j(u) V_j(v) 1\{L_j(u) = L_j(v)\} \quad (12)$$

Recall that we always assume the data (u, v) are normalized. For example, if the data are binary, then we have $u_i = \frac{1}{\sqrt{f_1}}$, $v_i = \frac{1}{\sqrt{f_2}}$. Hence the values $V_j(u)$ and $V_j(v)$ are small (and we need the term $\sqrt{f_1 f_2}$).

This estimator is again unbiased. Theorem 2 proves the mean the variance of $\hat{K}_{C,2}$.

Theorem 2

$$E(\hat{K}_{C,2}) = K_{C,2} \quad (13)$$

$$\begin{aligned} \text{Var}(\hat{K}_{C,2}) &= \frac{1}{k} \frac{f_1 f_2}{f_1 + f_2 - a} \left(\sum_{i=1}^D u_i^2 v_i^2 - \frac{(\sum_{i=1}^D u_i v_i)^2}{(f_1 + f_2 - a)} \right) \end{aligned} \quad (14)$$

Proof: See Appendix B. \square

Once we understand how to express $\hat{K}_{C,1}$ as an inner product, it should be easy to see that $\hat{K}_{C,2}$ can also be written as an inner product. Again, suppose $D = 4$, $L_j(u) = 2$, and $V_j(u) = 0.05$. We can consider an equivalent vector

$[0, 0.05\sqrt{f_1}, 0, 0]$. In other words, the difference between $\hat{K}_{C,1}$ and $\hat{K}_{C,2}$ is what value we should put in the nonzero location. Compared to $\hat{K}_{C,1}$, one advantage of $\hat{K}_{C,2}$ is that it only requires the permutations and thus eliminates the cost for conducting random projections.

As one would expect, the variance of $\hat{K}_{C,2}$ would be large if the data are heavy-tailed. However, when the data are appropriately normalized (e.g., via the TF-IDF transformation, or simply binarized), $\text{Var}(\hat{K}_{C,2})$ is actually quite small. Consider the extreme case when the data are binary, i.e., $u_i = \frac{1}{\sqrt{f_1}}$, $v_i = \frac{1}{\sqrt{f_2}}$, we have $\text{Var}(\hat{K}_{C,2}) = \frac{1}{k} (R - R^2)$, which is (considerably) smaller than $\text{Var}(\hat{K}_{C,1}) = \frac{1}{k} \{(1 + 2\rho^2) R - \rho^2 R^2\}$.

4.3 Experiment for Validation

To validate the theoretical results in Theorem 1 and Theorem 2, we provide a set of experiments in Figure 3. Two pairs of word vectors are selected: “A–THE” and “HONG–KONG”, from a chunk of web crawls. For example, the vector “HONG” is a vector whose i -th entry is the number of occurrences of the word “HONG” in the i -th document. For each pair, we apply the two proposed hashing algorithms to estimate $K_{C,1}$ and $K_{C,2}$. With sufficient repetitions (i.e., k), we can empirically compute the mean square errors ($\text{MSE} = \text{Var} + \text{Bias}^2$), which should match the theoretical variances if the estimators are indeed unbiased and the variance formulas, (11) and (14), are correct.

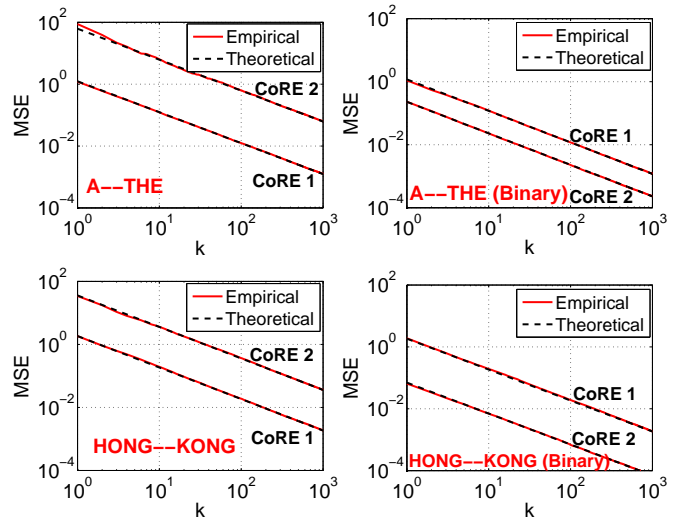


Figure 3: Mean square errors ($\text{MSE} = \text{Var} + \text{Bias}^2$) on two pairs of word vectors for validating Theorems 1 and 2. The empirical MSEs (solid curves) essentially overlap the theoretical variances (dashed curves), (11) and (14). When using the raw counts (left panels), the MSEs of $\hat{K}_{C,2}$ is significantly higher than the MSEs of $\hat{K}_{C,1}$. However, when using binarized data (right panels), the MSEs of $\hat{K}_{C,2}$ become noticeably smaller, as expected.

The number of word occurrences is a typical example of highly heavy-tailed data. Usually when text data are used in machine learning tasks, they have to be appropriately weighted (e.g., TF-IDF) or simply binarized. Figure 3 presents the results on the original data (raw counts) as well as the binarized data, to verify the formulas in Theorem 1 and Theorem 2, for $k = 1$ to 1000.

Indeed, the plots show that the empirical MSEs essentially overlap the theoretical variances. In addition, the MSEs of $\hat{K}_{C,2}$ is significantly larger than the MSEs of $\hat{K}_{C,1}$ on the raw data, as expected. Once the data are binarized, the MSEs of $\hat{K}_{C,2}$ become smaller, also as expected.

5 HASHING CORE KERNELS FOR SVM

In this section, we provide a set of experiments for using the hashed data as input for a linear SVM solver (LIBLINEAR). Our goal is to approximate the (nonlinear) CoRE kernels with linear kernels. In Section 4, we have explained how to express the estimators $\hat{K}_{C,1}$ and $\hat{K}_{C,2}$ as inner products by expanding the hashed data. With k permutations and k random projections, the number of nonzeros of the expanded data is precisely k . To reduce the dimensionality, we use only the lowest b bits of the locations [22]. In this study, we experiment with $b = 1, 2, 4, 8$.

Figure 4 presents the results on the **M-Rotate** dataset. As shown in Figure 1 and Table 1, using linear kernel can only achieve a test accuracy of 48%. This means, if we use random projections (or the variants, e.g., [21, 28]), which approximate inner products, then the best accuracy we can achieve would be about 48%. For this dataset, the performance of CoRE kernels (and resemblance kernel) is astonishing, as shown in Figure 2 and Table 2. Thus, we choose this dataset to demonstrate our proposed hashing algorithms combined with linear SVM can also approach the performance of (nonlinear) CoRE kernels.

To explain the procedure, we use the same examples as in Section 4. Suppose we apply k minwise hashing and k random projections on the data and we consider without loss of generality the data vector u . For the j -th projection and j -th minwise hashing, suppose $L_j(u) = 2, V_j(u) = 0.05, P_j(u) = 0.1$. Recall L_j and V_j are, respectively, the location and the value of the first nonzero entry after minwise hashing. P_j is the projected value obtained from random projection.

In order to use linear SVM to approximate kernel SVM with Type 1 CoRE kernel, for the above example, we expand the j -th hashed data as a vector $[0, 0.1, 0, 0]$ if $b = 2$, or $[0, 0.1]$ if $b = 1$. We then concatenate k such vectors to form a vector of length $2^b \times k$ (with exactly k nonzeros). Before we feed the expanded hashed data to LIBLINEAR, we normalize the vectors to have unit norm. The experimental results are presented in the left panels of Figure 4.

To approximate Type 2 CoRE kernel, we expand the j -th hashed data of u as $[0, 0.05\sqrt{f_1}, 0, 0]$ if $b = 2$, or $[0, 0.05\sqrt{f_1}]$ if $b = 1$, where f_1 is the number of nonzero entries in the original data vector u . Again, we concatenate k such vectors. The experimental results are presented in the middle panels of Figure 4.

To approximate resemblance kernel, we expand the j -th hashed data of u as $[0, 1, 0, 0]$ if $b = 2$ or $[0, 1]$ if $b = 1$ and we concatenate k such vectors.

The results in Figure 4 are exciting because linear SVM on the original data can only achieve an accuracy of 48%. Our proposed hashing methods + linear SVM can achieve $> 86\%$. In comparison, using only the original b -bit minwise hashing, the accuracy can still reach about 80%. Again, we should mention that other hashing algorithms which aim at approximating the inner product (such as random projections and variants) can at most achieve the same result as using linear SVM on the original data. This is the significant advantage of CoRE kernels.

6 DISCUSSIONS

There is a line of related work called *Conditional Random Sampling (CRS)* [19, 20] which was also designed for sparse non-binary data. Basically, the idea of CRS is to keep the first (smallest) k nonzero entries after applying one permutation on the data. [19, 20] developed the trick to construct an (essentially) equivalent random sample for each pair. CRS is naturally applicable to non-binary data and is capable of estimating any (linear) summary statistics, in static as well as dynamic (streaming) settings. In fact, the estimators developed for CRS can be (substantially) more accurate than the estimator for minwise hashing.

The major drawback of CRS is that the samples are not appropriately aligned. Consequently, CRS is not suitable for training linear SVM (or other applications which require the input data to be in a metric space). Our method has overcome this drawback. Of course, CRS can still be used in important scenarios such as estimating similarities during the re-ranking stage in LSH.

Why do we need to two types of CoRE kernels? While hashing Type 2 CoRE kernel is simpler because it requires only the random permutations, Table 2 shows that Type 1 CoRE kernel can often achieve better results than Type 2 CoRE kernel. Therefore, we develop hashing methods for both CoRE kernels, to provide users with more choices.

There are many promising extensions. For example, we can construct new kernels based on CoRE kernels (which currently do not have tuning parameters), by using the exponential function and introducing an additional tuning parameter γ , just like RBF kernel. This will allow more flexibility and potentially further improve the performance.

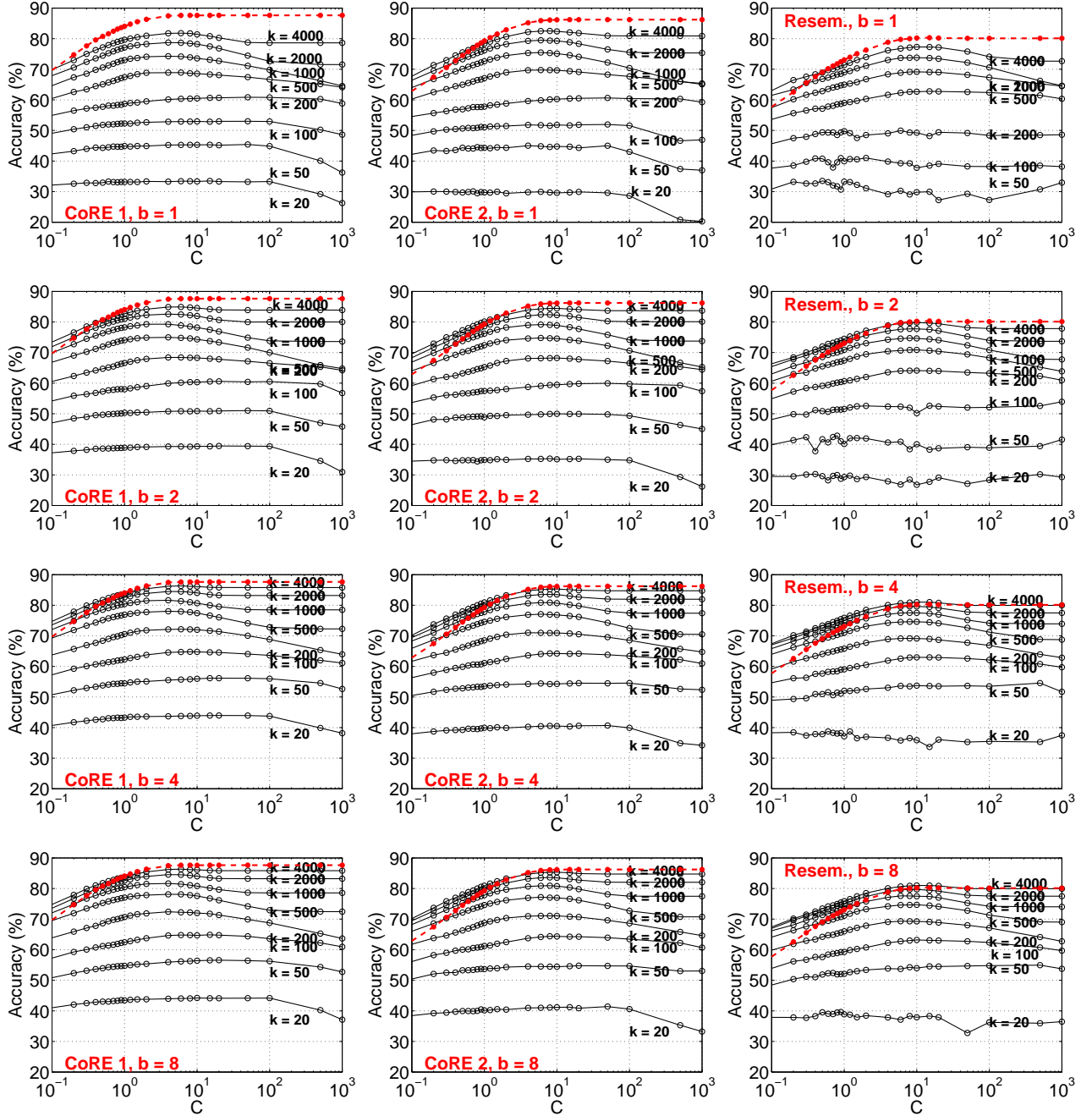


Figure 4: Test classification accuracies on the *M-Rotate* dataset using our proposed hashing methods and linear SVM (LIBLINEAR). The red (if color is available) dot curves are the results of kernel SVM on the original data (i.e., the same curves from Figure 2), using Type 1 CoRE kernel (left panels), Type 2 CoRE kernel (middle panels), and resemblance kernel (right panels), respectively. We apply both b -bit minwise hashing (with $b = 1, 2, 4, 8$) and random projections k times and feed the (expanded) hashed data to linear SVM.

Another interesting line of extensions would be applying other hashing algorithms on top of our generated hashed data. This is possible again because we can view our estimators as inner products and hence we can apply other hashing algorithms which approximate inner products on top of our hashed data. The advantage is the potential further data compression. Another advantage would be in the context of sublinear time approximate near neighbor search (when the target similarity is the CoRE kernels).

For example, we can apply another layer of random projections on top of the hashed data and then store the signs of the new projected data [6, 10]. These signs, which are bits, provide good indexing & space partitioning capability to allow sublinear time approximate near neighbor search under the framework of *locality sensitive hashing (LSH)* [13]. This way, we can search for near neighbors in the space of CoRE kernels (instead of the space of inner products).

In addition, we expect that our work will inspire new research on the development of more efficient (b -bit) minwise hashing methods when the size of the space (i.e., D) is not too large and the data are not necessarily extremely sparse. Traditionally, minwise hashing has been used as a data size/dimensionality reduction tool, typically for very large D (e.g., 2^{64}). Readers perhaps have noticed that, in our paper, (b -bit) minwise hashing could be utilized as a *data expansion* tool in order to apply efficient linear algorithms. When D is not very large, many aspects of the algorithms such as pseudo-random number generation would be quite different and new research may be necessary.

7 CONCLUSION

Current popular hashing methods, such as random projections and variants, often focus on approximating inner products and large-scale linear classifiers (e.g., linear SVM). However, linear kernels often do not achieve good performance. In this paper, we propose two types of nonlinear CoRE kernels which outperform linear kernels, sometimes by a large margin, on sparse non-binary data (which are common in practice). Because CoRE kernels are nonlinear, we accordingly develop basic hash methods to approximate CoRE kernels with linear kernels. The hashed data can be fed into highly efficient linear classifiers. Our experiments confirm the findings. We expect this work will inspire a new line of research on kernel learning, hashing algorithms, and large-scale learning.

ACKNOWLEDGEMENT

The research of Ping Li is partially supported by NSF-III-1360971, NSF-Bigdata-1419210, ONR-N00014-13-1-0764, and AFOSR-FA9550-13-1-0137.

A Proof of Theorem 1

To compute the expectation and variance of the estimator $\hat{K}_{C,1} = \frac{1}{k} \sum_{j=1}^k P_j(u)P_j(v)1\{L_j(u) = L_j(v)\}$, we need the first two moments of $P_j(u)P_j(v)1\{L_j(u) = L_j(v)\}$. The first moment is

$$\begin{aligned} E[P_j(u)P_j(v)1\{L_j(u) = L_j(v)\}] \\ = E[P_j(u)P_j(v)] \Pr(L_j(u) = L_j(v)) = \rho R \end{aligned}$$

which implies that $E(\hat{K}_{C,1}) = K_{C,1} = \rho R$. The second moment is

$$\begin{aligned} E[P_j^2(u)P_j^2(v)1\{L_j(u) = L_j(v)\}] \\ = E[P_j^2(u)P_j^2(v)] \Pr(L_j(u) = L_j(v)) \\ = (1 + 2\rho^2) \rho R \end{aligned}$$

Here, we have used the result in the prior work [21]: $E[P_j^2(u)P_j^2(v)] = 1 + 2\rho^2$. Therefore, the variance is

$$\text{Var}(\hat{K}_{C,1}) = \frac{1}{k} \{(1 + 2\rho^2) R - \rho^2 R^2\}$$

This completes the proof.

B Proof of Theorem 2

We need the first two moments of the estimator $\hat{K}_{C,2} = \frac{1}{k} \sum_{j=1}^k V_j(u)V_j(v)1\{L_j(u) = L_j(v)\}\sqrt{f_1 f_2}$

Because

$$\begin{aligned} E[V_j(u)V_j(v)1\{L_j(u) = L_j(v)\}] \\ = E[V_j(u)V_j(v)1\{L_j(u) = L_j(v)\} | L_j(u) = L_j(v)] \\ \times \Pr(L_j(u) = L_j(v)) \\ = \frac{\sum_{i=1}^D u_i v_i}{a} R = \rho \frac{1}{f_1 + f_2 - a} \end{aligned}$$

we know

$$E(\hat{K}_{C,2}) = \frac{1}{k} \sum_{j=1}^k \rho \frac{\sqrt{f_1 f_2}}{f_1 + f_2 - a} = K_{C,2}$$

and

$$\begin{aligned} E[V_j^2(u)V_j^2(v)1\{L_j(u) = L_j(v)\}] \\ = E[V_j^2(u)V_j^2(v)] \Pr(L_j(u) = L_j(v)) \\ = \frac{\sum_{i=1}^D u_i^2 v_i^2}{a} R = \frac{\sum_{i=1}^D u_i^2 v_i^2}{f_1 + f_2 - a} \end{aligned}$$

Therefore,

$$\begin{aligned} \text{Var}(\hat{K}_{C,2}) \\ = \frac{1}{k} \frac{f_1 f_2}{f_1 + f_2 - a} \left(\sum_{i=1}^D u_i^2 v_i^2 - \frac{(\sum_{i=1}^D u_i v_i)^2}{(f_1 + f_2 - a)} \right) \end{aligned}$$

This completes the proof.

References

- [1] L. Bottou. <http://leon.bottou.org/projects/sgd>.
- [2] L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, editors. *Large-Scale Kernel Machines*. The MIT Press, Cambridge, MA, 2007.
- [3] A. Z. Broder. On the resemblance and containment of documents. In *the Compression and Complexity of Sequences*, pages 21–29, Positano, Italy, 1997.
- [4] G. Buehrer and K. Chellapilla. A scalable pattern mining approach to web graph compression with communities. In *WSDM*, pages 95–106, Stanford, CA, 2008.
- [5] T. Chandra, E. Ie, K. Goldman, T. L. Llinares, J. McFadden, F. Pereira, J. Redstone, T. Shaked, and Y. Singer. Sibyl: a system for large scale machine learning. Technical report, 2010.
- [6] M. S. Charikar. Similarity estimation techniques from rounding algorithms. In *STOC*, pages 380–388, Montreal, Quebec, Canada, 2002.
- [7] F. Chierichetti, R. Kumar, S. Lattanzi, M. Mitzenmacher, A. Panconesi, and P. Raghavan. On compressing social networks. In *KDD*, pages 219–228, Paris, France, 2009.
- [8] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.
- [9] D. Fetterly, M. Manasse, M. Najork, and J. L. Wiener. A large-scale study of the evolution of web pages. In *WWW*, pages 669–678, Budapest, Hungary, 2003.
- [10] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of ACM*, 42(6):1115–1145, 1995.
- [11] S. Gollapudi and A. Sharma. An axiomatic approach for result diversification. In *WWW*, pages 381–390, Madrid, Spain, 2009.
- [12] M. R. Henzinger. Finding near-duplicate web pages: a large-scale evaluation of algorithms. In *SIGIR*, pages 284–291, 2006.
- [13] P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *STOC*, pages 604–613, Dallas, TX, 1998.
- [14] N. Jindal and B. Liu. Opinion spam and analysis. In *WSDM*, pages 219–230, Palo Alto, California, USA, 2008.
- [15] T. Joachims. Training linear svms in linear time. In *KDD*, pages 217–226, Pittsburgh, PA, 2006.
- [16] H. Larochelle, D. Erhan, A. C. Courville, J. Bergstra, and Y. Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *ICML*, pages 473–480, Corvallis, Oregon, 2007.
- [17] P. Li. Abc-boost: Adaptive base class boost for multi-class classification. In *ICML*, pages 625–632, Montreal, Canada, 2009.
- [18] P. Li. Robust logitboost and adaptive base class (abc) logitboost. In *UAI*, 2010.
- [19] P. Li and K. W. Church. Using sketches to estimate associations. In *HLT/EMNLP*, pages 708–715, Vancouver, BC, Canada, 2005.
- [20] P. Li, K. W. Church, and T. J. Hastie. Conditional random sampling: A sketch-based sampling technique for sparse data. In *NIPS*, pages 873–880, Vancouver, BC, Canada, 2006.
- [21] P. Li, T. J. Hastie, and K. W. Church. Very sparse random projections. In *KDD*, pages 287–296, Philadelphia, PA, 2006.
- [22] P. Li, A. Shrivastava, J. Moore, and A. C. König. Hashing algorithms for large-scale learning. In *NIPS*, Granada, Spain, 2011.
- [23] M. Najork, S. Gollapudi, and R. Panigrahy. Less is more: sampling the neighborhood graph makes salsa better and faster. In *WSDM*, pages 242–251, Barcelona, Spain, 2009.
- [24] S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated sub-gradient solver for svm. In *ICML*, pages 807–814, Corvallis, Oregon, 2007.
- [25] S. Tong. Lessons learned developing a practical large scale machine learning system. <http://googleresearch.blogspot.com/2010/04/lessons-learned-developing-practical.html>, 2008.
- [26] T. Urvoy, E. Chauveau, P. Filoche, and T. Lavergne. Tracking web spam with html style similarities. *ACM Trans. Web*, 2(1):1–28, 2008.
- [27] J. Wang, J. Yang, K. Yu, F. Lv, T. S. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *CVPR*, pages 3360–3367, San Francisco, CA, 2010.
- [28] K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg. Feature hashing for large scale multitask learning. In *ICML*, pages 1113–1120, 2009.
- [29] K. Yu, T. Zhang, and Y. Gong. Nonlinear learning using local coordinate coding. In *NIPS*, Vancouver, BC, Canada, 2009.

Efficient Sparse Recovery via Adaptive Non-Convex Regularizers with Oracle Property

Ming Lin *

Department of Automation
Tsinghua University
Beijing, P.R. China 100084

Rong Jin

Computer Science and Engineering
Michigan State University
East Lansing, MI, USA 48823

Changshui Zhang

Department of Automation
Tsinghua University
Beijing, P.R. China 100084

Abstract

The main shortcoming of sparse recovery with a convex regularizer is that it is a biased estimator and therefore will result in a suboptimal performance in many cases. Recent studies have shown, both theoretically and empirically, that non-convex regularizer is able to overcome the biased estimation problem. Although multiple algorithms have been developed for sparse recovery with non-convex regularization, they are either computationally demanding or not equipped with the desired properties (i.e. optimal recovery error, selection consistency and oracle property). In this work, we develop an algorithm for efficient sparse recovery based on proximal gradient descent. The key feature of the proposed algorithm is introducing adaptive non-convex regularizers whose shrinking threshold vary over iterations. The algorithm is compatible with most popular non-convex regularizers, achieves a geometric convergence rate for the recovery error, is selection consistent, and most importantly has the oracle property. Based on the proposed framework, we suggest to use a so-called ACCQ regularizer, which is equivalent to zero proximal projection gap adaptive hard-thresholding. Experiments with both synthetic data sets and real images verify both the efficiency and effectiveness of the proposed method compared to the state-of-the-art methods for sparse recovery.

1 INTRODUCTION

Inspired by the seminal work of compressive sensing (Candès et al., 2006), numerous algorithms have been

developed to recover a sparse vector from its linear low dimensional measurement. Most of these algorithms can be classified into two categories: greedy methods and optimization-based methods. Greedy methods aggressively select the support set as they recover the target sparse vector ((Tropp and Wright, 2010) and references therein). Although they are computationally efficient, the greedy methods are usually sensitive to noise especially when the target signal is not exactly sparse. Optimization-based methods, on the other hand, are known to be more robust to noise but at the price of a higher computational cost (Zou and Hastie, 2005; Rosenbaum and Tsybakov, 2010; Xu et al., 2010).

Most optimization-based methods cast sparse recovery into convex optimization problems. The most well known algorithms in this category are LASSO (Tibshirani, 1996; Efron et al., 2004) and Dantzig selector (Candes and Tao, 2007). A main drawback of convex optimization based methods for sparse recovery is that they are biased estimators, i.e. the solutions found by the convex optimization based methods do not have the *oracle property* (Fan and Li, 2001; Zhang and Zhang, 2011), which is sometime referred to as *Lasso bias* (Zhang and Zhang, 2011). We note that there are two different versions of oracle property used in the literature: the asymptotical one that examines the oracle property with the number of measurements going to infinity (Fan and Li, 2001) and the finite sample one (Zhang and Zhang, 2011) that examines the oracle property with a finite number of measurements. In this study, we will use the finite sample version of oracle property.

It was first suggested in (Fan and Li, 2001) that the Lasso bias can be corrected by a non-convex regularizer. Several theory and algorithms have been developed recently for sparse recovery using concave regularizers (Zhang and Zhang, 2011; Zhang, 2012; Loh and Wainwright, 2013), and their effectiveness for sparse recovery has been verified empirically by several recent studies (Xiang et al., 2013; Gong et al., 2013a; Ochs et al., 2013). Despite the appealing result, it remains to be challenging as how to efficiently solve the optimization problem with non-convex regular-

Ming Lin and Changshui Zhang are from State Key Laboratory on Intelligent Technology and Systems, Tsinghua National Laboratory for Information Science and Technology (TNList), Department of Automation, Tsinghua University, Beijing, P.R. China 100084.

izer.

Zhang (2012) proposed a multi-stage algorithm that relaxes a non-convex optimization problem into a sequence of convex optimization problems with weighted ℓ_1 regularizers. Besides the recovery error, the author also showed in (Zhang, 2012) that the solution found by multi-stage algorithm satisfies the oracle property when the non-zero entries in the target sparse vector are sufficiently large. The main shortcoming of the multi-stage algorithm is its potentially high computational cost as it needs to solve a sequence of ℓ_1 regularized optimization problems. (Zhao-ran Wang, 2013) relax this problem by computing approximate solution at each stage, but still require multiple stages thus is not very efficient. Several proximal gradient descent methods have been proposed for non-convex regularizers (Gong et al., 2013b; Loh and Wainwright, 2013) that enjoy higher computational efficiency than the multi-stage algorithm. However, it is unclear if the solutions found by these algorithms will be unbiased estimators and have the oracle property, the key reason for using a non-convex regularizer.

In this work, we propose an algorithm for sparse recovery using adaptive non-convex regularizer to develop efficient algorithms with all the desired properties above. The proposed framework, on one hand, enjoys the high computational efficiency and achieves a linear convergence rate in the recovery error as some of the proximal gradient descent methods do. On the other hand, like the multi-stage algorithm, the proposed framework is able to find a sparse solution with optimal recovery error, selection consistency, and oracle property under appropriate conditions. The key feature introduced by the proposed framework is introducing *adaptive concave regularizers* whose shrinking-threshold vary over iterations. It is the introduction of the adaptive concave regularizer that allows us to effectively remove the noise and identify the support set, leading to high computational efficiency and a solution with optimal recovery error and oracle property.

Although the proposed algorithm is compatible with most popular non-convex regularizers, via a more deep examination, we find that the type of non-convex regularizer is not important at all. What really matters is the so-called *proximal projection gap* that will be defined later. This gap determines the bias of regularizer in sparse estimation. Based on this discovery, we propose to use a so-called ACCQ regularizer, whose proximal projection gap is zero. From optimization viewpoint, the ACCQ regularizer is equivalent to one kind of hard-thresholding algorithms with adaptive threshold. The ACCQ regularizer is the only regularizer whose projection gap is zero, thus is superior than other alternatives.

The rest of this paper is organized as following. Section 2 reviews the related work. Section 3 describes the proposed algorithm for sparse recovery. Section 4 analyzes theoretic

cal properties. Experimental results with both synthesized and real data sets are summarized in Section 5. Section 6 encloses our study with open questions.

2 RELATED WORK

We briefly review the related work on sparse recovery, with focus on non-convex regularizer. More complete references on the related subject can be found in (Tropp and Wright, 2010), (Davenport et al., 2011) and (Zhang and Zhang, 2011).

Most sparse recovery methods are based on ℓ_1 regularization. The most well algorithm is LASSO (Tibshirani, 1996; Efron et al., 2004). Numerous algorithms have been developed to solve LASSO related optimization problem efficiently (Beck and Teboulle, 2009; Foucart, 2012). It has been shown that ℓ_1 regularization can be solved efficiently with a linear convergence (up to stochastic tolerance) (Xiao and Zhang, 2012; Agarwal et al., 2012). A main problem with LASSO is that it is a biased estimator. In particular, LASSO is unable to perfectly recover the solution of oracle Least Square Estimation (LSE), a property that is usually referred to as *oracle property*. We emphasize that the LASSO bias is not an artifact of analysis, and it does show up noticeably in the recovery error, according to our empirical study as well as others (Zhang, 2012; Zou, 2006). It was pointed out in (Fan and Li, 2001; Zhang and Zhang, 2011) that LASSO bias also exists in other convex regularizers.

Multiple non-convex regularizers have been proposed to address the bias of convex regularizers, including Geman Penalty (GP) (Geman and Yang, 1995; Trzasko and Manduca, 2009), SCAD (Fan and Li, 2001), Log Sum Penalty (LSP) (Candes et al., 2008), ℓ_q norm (Foucart and Lai, 2009), Minimax Concave Penalty (MCP) (Zhang, 2010a), Capped- ℓ_1 norm (Loh and Wainwright, 2013). Various algorithms have been developed to find local optimal for non-convex regularizers (Zhang and Zhang, 2011) and references therein). It is however unclear if the local solutions found by these algorithms have the desired properties (i.e. the optimal recovery error and the oracle property). Only a handful algorithms that achieve the desired properties, including the multi-stage algorithm (Zhang, 2010b), adaptive LASSO (Zou, 2006) that can be shown as a special case of multi-stage algorithm and achieve the asymptotical oracle property, and the forward and backward regression scheme (FOBA) (Zhang, 2011), based on adaptive regularization, also finds the solutions with all the desired properties. FOBA is guaranteed to terminate within $O(s)$ iterations, where s is the sparsity. The main limitation of FOBA is that it is unclear if the oracle property recovery error of their algorithm can achieve a linear convergence rate. Each iteration of FOBA is an optimization problem therefore is not efficient. Ji Liu (2013) propose an variant of FOBA but

they still suffer the same problem. The FOBA and its variants make different assumptions that is complementary to our analysis. Although the multi-stage algorithm achieves a linear convergence, it requires to solve a weighted ℓ_1 regularization problem that can be computationally costly when the dimension of data is high.

3 SPARSE RECOVERY BY ADAPTIVE NON-CONVEX REGULARIZER

In this section, we first introduce the background materials and notations for sparse recovery. We then present our algorithm and its main theoretical property. The sketch of proofs is given at the end of this section.

3.1 BACKGROUNDS AND NOTATIONS

Let A be an $n \times d$ design matrix and $\mathbf{y} \in \mathbb{R}^n$ be response vector satisfying

$$\mathbf{y} = A\mathbf{x}_* + \mathbf{z}, \quad (1)$$

where $\mathbf{x}_* \in \mathbb{R}^d$ is the s -sparse vector to be recovered and $\mathbf{z} \in \mathbb{R}^n$ is a noise vector. We assume that each element $[A^\top \mathbf{z}]_i$ follows a subgaussian distribution with $\|[A^\top \mathbf{z}]_i\|_{\psi_2} \leq \sigma\sqrt{n}, i = 1, \dots, d$, where σ indicates the noise level in \mathbf{z} and $\|\cdot\|_{\psi_2}$ is Orlicz norm (Koltchinskii, 2011). Using the property of subgaussianity, we have, with a high probability $(1 - d^{-3})$,

$$\|[A\mathbf{z}]_i\|_\infty \leq 2\sigma\sqrt{n \log d} \quad (2)$$

For the rest of the paper, we will simply assume condition (2) holds.

Following (Candes and Tao, 2005), we assume A satisfies *Restricted Isometric Property* (R.I.P.) defined as follows.

Definition 1 (*Restricted Isometric Property*). A matrix A satisfies δ_s -R.I.P. condition, if there exists a positive constant δ_s such that for all s -sparse vector \mathbf{x} ,

$$(1 - \delta_s) \|\mathbf{x}\|_2^2 \leq \frac{1}{n} \|A\mathbf{x}\|_2^2 \leq (1 + \delta_s) \|\mathbf{x}\|_2^2. \quad (3)$$

A is called $\delta_{s,s}$ -restricted orthogonal, if there exists a positive constant $\delta_{s,s}$ such that for any two s -sparse vector \mathbf{u}, \mathbf{v} whose support sets are disjoint,

$$\frac{1}{n} |\langle A\mathbf{u}, A\mathbf{v} \rangle| \leq \delta_{s,s} \|\mathbf{u}\|_2 \|\mathbf{v}\|_2. \quad (4)$$

Small δ_s and $\delta_{s,s}$ indicate that A is approximately isometric on sparse subspace and any two set of s columns are approximately orthogonal if they are disjoint. In the rest of this paper, we will say A is δ -R.I.P. when both (3) and (4) are satisfied with $\delta = \max\{\delta_{2s}, \delta_{s,s}\}$.

Algorithm 1 Proximal Gradient Descent With Adaptive Capped Concave Quadratic (ACCQ) Regularizer

Input: the size of target vector $R \geq \|\mathbf{x}_*\|_2$, design matrix A , measurements \mathbf{y} , threshold θ , shrinking parameter $q \in (0, 1)$, and number of iterations T

- 1: **Initialization:** $\mathbf{x}_1 = 0$
- 2: **for** $t = 1$ to T **do**
- 3: Compute τ_t by $\tau_t = Rq^{t-1} + \theta$
- 4: Compute $\hat{\mathbf{x}}_{t+1}$ by $\hat{\mathbf{x}}_{t+1} = \mathbf{x}_t - \nabla L(\mathbf{x}_t)$
- 5: Update \mathbf{x}_{t+1} using (8)
- 6: **end for**

Output: \mathbf{x}_{T+1}

There are other alternative conditions for sparse recovery which are more general than R.I.P, such as restricted eigenvalue condition (Bickel et al., 2009). A complete list of conditions for sparse recovery and their comparison can be found (Van De Geer and Bühlmann, 2009). We choose R.I.P condition due to its simplicity, and extension to more general cases will be studied in the future. From now on we will assume that A is δ -R.I.P.. In experiments we generate A from random Gaussian distribution, which is widely known to obey the R.I.P. with a high probability.

To recover the sparse vector \mathbf{x}_* , a common approach is to minimize the regularized empirical loss

$$\min_{\mathbf{x}} \frac{1}{2n} \|\mathbf{y} - A\mathbf{x}\|_2^2 + \Omega(\mathbf{x}), \quad (5)$$

where $\Omega(\mathbf{x})$ is a regularizer that controls the sparsity of the solution. To remove the LASSO bias, a non-convex regularizer is used for $\Omega(\mathbf{x})$, leading to a non-convex optimization problem that is not only difficult to solve numerically and but also challenging to analyze the theoretical properties for the found solution.

The following notation will be used throughout this paper. For a vector \mathbf{x} , we denote by $[\mathbf{x}]_i$ the i -th entry of \mathbf{x} , by $|\mathbf{x}|_i$ the absolute value of $[\mathbf{x}]_i$, by $[\mathbf{x}]_A$ the subvector of \mathbf{x} that only includes the elements in the index set $A \subseteq [d]$, and by $\lambda_{\min}(\mathbf{x})$ the minimum absolute value of the non-zero entries in \mathbf{x} . We will use $\text{supp}(\mathbf{x})$ for the support set for a vector \mathbf{x} . We will use $\|\mathbf{x}\|_2$, $\|\mathbf{x}\|_1$, and $\|\mathbf{x}\|_\infty$ to represent the ℓ_2 , ℓ_1 , and ℓ_∞ norm of vector \mathbf{x} . We will denote by S_* the support set of \mathbf{x}_* and by S_t the support set for \mathbf{x}_t .

3.2 PROXIMAL GRADIENT DESCENT USING ADAPTIVE CAPPED CONCAVE QUADRATIC (ACCQ) REGULARIZER

The proposed framework essentially follows the proximal gradient descent method that has been widely used in convex optimization. At each iteration, we first obtain an aux-

iliary solution $\hat{\mathbf{x}}_t$ by

$$\begin{aligned} \hat{\mathbf{x}}_t &= \mathbf{x}_t - \nabla L(\mathbf{x}_t) \\ \text{where } L(\mathbf{x}) &= \frac{1}{2n} \|\mathbf{y} - A\mathbf{x}\|_2^2. \end{aligned} \quad (6)$$

The updated solution \mathbf{x}_{t+1} is then given by

$$\mathbf{x}_{t+1} = \arg \min_{\mathbf{x}} \frac{1}{2} \|\mathbf{x} - \hat{\mathbf{x}}_t\|_2^2 + \Omega_t(\mathbf{x}) \quad (7)$$

where the regularizer Ω_t has a subscript t and therefore varies from trial to trial.

To ensure Eq. (7) leading to an unbiased sparse estimation, we make some assumptions on the adaptive regularizer $\Omega_t(\mathbf{x})$. The type of $\Omega_t(\mathbf{x})$ is not important at all. We only care about its shrinking strategy in step Eq. (7).

Assumption 1. Define variables τ_t and constant α . The adaptive non-convex regularizer $\Omega_t(\mathbf{x})$ in Eq. (7) shrinks $\hat{\mathbf{x}}_t$ in the following way:

- For $|\hat{\mathbf{x}}_t|_i < \tau_t$, $[\mathbf{x}_{t+1}]_i = 0$.
- For $|\hat{\mathbf{x}}_t|_i > \tau_t + \alpha$, $[\mathbf{x}_{t+1}]_i = [\hat{\mathbf{x}}_t]_i$.
- For $\tau_t \leq |\hat{\mathbf{x}}_t|_i \leq \tau_t + \alpha$, $0 < [\mathbf{x}_{t+1}]_i < [\hat{\mathbf{x}}_t]_i$.

The τ_t is a threshold parameter that adaptively shrinks over iterations. We will describe the updating rule of τ_t later in Eq. (9). α is called *proximal projection gap*. In Assumption 1, if the intermedia solution $|\hat{\mathbf{x}}_t|_i$ is outside $[\tau_t, \tau_t + \alpha]$, the proximal projection of Ω_t is equivalent to hard-thresholding. Otherwise $|\hat{\mathbf{x}}_t|_i$ is projected onto $(0, |\hat{\mathbf{x}}_t|_i)$, whose value depends on the specific regularizer being used. The hard-thresholding is the key to unbiased estimation, which is only possible when using non-convex regularizer. The proximal projection gap α reflects the non-convexity of $\Omega_t(\mathbf{x})$ in the proximal projection. For convex regularizer, α is infinity by definition because they are soft-thresholding methods that never do hard-thresholding. For a particular non-convex regularizer, we naturally prefer small α to avoid involving bias as much as possible.

The shrinking strategy of non-convex regularizer is very similar to greedy algorithms. The following concavity assumption distinguish non-convex methods from greedy methods, which at the same time build a bridge of the two realms.

Assumption 2. $\Omega_t(\mathbf{x})$ is concave in \mathbf{x} :

$$\Omega_t(\mathbf{x}_1) - \Omega_t(\mathbf{x}_2) \leq \langle \partial \Omega_t(\mathbf{x}_2), \mathbf{x}_1 - \mathbf{x}_2 \rangle, \|\partial \Omega_t(\mathbf{x})\| \leq \tau_t,$$

where $\partial \Omega_t(\mathbf{x})$ is the subgradient.

Most popular static non-convex regularizer and their adaptive variants fit Assumption 1 and Assumption 2, with different τ_t and α . We list a few of them in Table 1. We notice that in Table 1, most regularizers' α is not zero. Although our theory could deal with non-zero α , we naturally hope

there is a regularizer that doesn't need to suffer this gap α , which results in a better estimation. Inspired by this observation, we introduce *Adaptive Capped Concave Quadratic* regularizer (ACCQ), defined in the last row of Table 1. Clearly, this regularizer is the only hard-thresholding regularizer whose proximal projection gap is zero. Its proximal projection is given by:

$$[\mathbf{x}_{t+1}]_i = \begin{cases} [\hat{\mathbf{x}}_t]_i & |\hat{\mathbf{x}}_t|_i \geq \tau_t \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

This specific adaptive hard-thresholding strategy allows us to correct the LASSO bias and consequentially achieve the oracle property when the signals in the target vectors are strong.

In the rest of this paper, we only focus on ACCQ and its recovery properties. For regularizers with $\alpha > 0$, all of the following theorems hold true except an extra α in the upper bound. Therefore they are always suboptimal compared with ACCQ.

Threshold parameter τ_t is determined by the following equation

$$\tau_t = Rq^{t-1} + \theta. \quad (9)$$

- Threshold parameter $\theta > 0$ determines the lower bound for τ_t . It is introduced to ensure that the regularization is strong enough to overcome the noise.
- Shrinking parameter $q \in (0, 1)$ controls the speed of shrinkage. The idea of using a shrinking regularizer is motivated by a simple observation: as we go through the iteration t , we expect the solution \mathbf{x}_t will approach the target vector \mathbf{x}_* with a smaller error. As a result, only a smaller regularization is needed to overcome the noise caused by the recovery error. We note that similar shrinking strategy has been used in sparse recovery with the ℓ_1 regularizer (Xiao and Zhang, 2012).

Algorithm 1 gives the details for the proposed algorithm.

Remark 1 In practice, the settings of q and θ is robust, as suggested by our theorems and experiments. We suggest to set $0.9 \leq q < 1$ and $\theta \in [O(\sigma/\sqrt{n}), \lambda_{\min}(\mathbf{x}_*)]$. For example, $q = 0.95$, $\theta = 0.005$ usually satisfies our assumption and works well in practice.

Remark 2 It is interesting to compare Algorithm 1 with greedy hard-thresholding algorithms like GraDeS. GraDeS keeps exactly s entries at each iteration, even if the smallest s -th entry contains large noise. The proposed algorithm gradually collects entries according to their magnitude and current estimation uncertainty. It doesn't keep entries that contain large noise, so at the beginning of each iteration, it will keep less than s entries. Another greedy algorithm is OMP, which greedy select entries then keep them as support set. OMP must ensure that it always collect right entry

Table 1: Adaptive Regularizers And Proximal Projection Gap

Name	$\Omega_t(\mathbf{x}_i)$	Gap α
adaptive ℓ_1 norm	$\tau_t \lambda \mathbf{x}_i $	∞
adaptive capped ℓ_1 norm	$\tau_t \lambda \min(\mathbf{x}_i , \theta) \quad \theta > 0$	$\tau_t(\theta - \lambda)$
adaptive MCP	$\tau_t \lambda \int_0^{ \mathbf{x}_i } \min(1, \frac{[\theta \lambda - x]_+}{(\theta - 1)\lambda}) dx \quad \theta \geq 2$	$\tau_t(\theta - 1)\lambda$
ACCQ	$\Omega_t^{\text{CCQ}}(\mathbf{x}) = \sum_{i=1}^d \begin{cases} -\frac{1}{2}(\mathbf{x} _i - \tau_t)^2 + \frac{1}{2}\tau_t^2 & \mathbf{x} _i < \tau_t \\ \frac{1}{2}\tau_t^2 & \text{otherwise} \end{cases}$	0

at each iteration, otherwise it will fail definitely. The propose algorithm adaptively throw out entries that is collected in the previous iterations. This strategy is clealy much more robst against noise.

3.3 MAIN THEORETICAL RESULTS

The following theorem shows that the recovery error for Algorithm 1 is reduced exponentially, and all the intermediate solutions are $2s$ -sparse.

Theorem 1. Assume \mathbf{x}_* is s -sparse, and $6\delta < 1$. Set parameter q , and θ in Algorithm 1 as

$$q = \max\left(3\delta, 2\sqrt{\frac{\delta}{1-2\delta}}\right), \quad \theta = 2\sigma\sqrt{\frac{\log d}{n}} \quad (10)$$

Let $\mathbf{x}_1, \dots, \mathbf{x}_T$ be the sequence of solutions output from algorithm 1. Let S_t be the support set for \mathbf{x}_t and let S_* be the support set for \mathbf{x}_* . We have

$$|S_t \setminus S_*| \leq s, \quad \|\mathbf{x}_t - \mathbf{x}_*\|_2 \leq Rq^{t-1} + \frac{4\sigma}{1-q}\sqrt{\frac{s \log d}{n}} \quad (11)$$

First, as revealed by Theorem 1, the recovery error will converge geometrically to $O(\sigma\sqrt{s \log d/n})$, which has shown to be minimax optimal in (Raskutti et al., 2009). Second, as indicated in (11), all the intermediate solutions are at most $2s$ -sparse, making it computational appealing as our algorithm only needs to maintain a vector of no more than $2s$ non-zero entries. This is similar to iterative hard thresholding algorithms (e.g. (Garg and Khandekar, 2009)). Finally, the linear convergence takes place when $\delta \leq 1/6$, which worse than some other conditions on δ (e.g. (Garg and Khandekar, 2009)). We believe that this condition is improvable by more carefully tuning the inequalities in our analysis via similar techniques demonstrated in (Garg and Khandekar, 2009).

Next, we will show that the solution found by Algorithm 1 is selection consistent and has the oracle property if $\lambda_{\min}(\mathbf{x}_*)$, the smallest absolute value for the non-zero entries in \mathbf{x}_* , is larger than $O(\sigma\sqrt{\log d/n})$. To this end, we first define the solution of Least Square Oracle (LSE) estimation.

Definition 2 (Least Square Oracle (LSE)). *The Least Square Oracle estimation is the least square estimation of \mathbf{x}_* by assuming that the support set S_* of \mathbf{x}_* is provided. The LSE solution \mathbf{x}_o is given by*

$$\mathbf{x}_o = \mathbf{x}_{S_*} = (A_{S_*}^\top A_{S_*})^{-1} A_{S_*}^\top \mathbf{y}.$$

Definition 3 (Selection Consistent and Oracle Property). *An estimator is selection consistent if the estimated solution $\hat{\mathbf{x}}$ satisfies $\text{supp}(\hat{\mathbf{x}}) = \text{supp}(\mathbf{x}_*)$, and has the oracle property if $\hat{\mathbf{x}} = \mathbf{x}_o$.*

We note that since the solution of oracle LSE \mathbf{x}_o is obtained without using any regularizer, the oracle property (i.e. $\hat{\mathbf{x}} = \mathbf{x}_o$) essentially ensures that the sparse recovery algorithm will not be biased by the regularizer, of course under the assumption that $\lambda_{\min}(\mathbf{x}_*)$ is sufficiently large. We note that the early definition of oracle property (e.g. (Fan and Li, 2001)) requires $\hat{\mathbf{x}}_{S_*} - \mathbf{x}_o$ converge to a Gaussian random vector when the number of measurements n goes to infinity, which is weaker than the finite sample version defined above.

Theorem 2. Assume $6\delta < 1$ and with q and θ set in (10). Define

$$t_0 = \log_2 \left(\frac{\max(R, 1)}{\sigma} \sqrt{\frac{\log d}{n}} \right)$$

Then, we have $S_t = S_*$ for $t > t_0$ (i.e. selection consistency) and

$$\|\mathbf{x}_t - \mathbf{x}_o\|_2 \leq \left(\frac{3\delta}{1-3\delta} \right)^{(t-t_0)/2} \frac{8\sigma}{1-q} \sqrt{\frac{s \log d}{n}}$$

if

$$\lambda_{\min}(\mathbf{x}_*) \geq \frac{4\sigma}{1-\delta} \sqrt{2 \frac{\log d}{n}} \quad (12)$$

and

$$s \leq \frac{1}{50\delta} (1-q)^2 (2 - \frac{1}{1-\delta}) \quad (13)$$

As revealed by the above theorem, \mathbf{x}_o , the solution of oracle LSE, can be perfectly recovered by Algorithm 1 with sufficiently large number of iterations, provided (i) the non-zero entries in \mathbf{x}_* are sufficiently large, and (ii) the RIP constant δ is sufficiently small.

Remark 1 Eq. (13) is essentially a particular form of *Generalized Uncertainty Principle* (GUP) (Candès et al., 2006). GUP claims that for any compress sensing methods, the number s of non-zeros elements in \mathbf{x}_* couldn't be larger than half of the number n of frequency sampling, i.e., $s \leq 0.5n$. Otherwise there is no algorithm could recover the sparse signal. In Eq. (13), generally speaking, $\delta = O(1/n)$. So Eq. (13) is claiming that s should be smaller than

$$s \leq cn,$$

where c is a constant. The constant c given by Eq. (13) is slightly loose than $1/2$ which is proven to be optimal in GUP. We believe this could be refined by carefully tuning the inequalities in our analysis.

Remark 2 $\lambda_{\min}(\mathbf{x}_*)$ in Eq. (12) doesn't contain \sqrt{s} thanks to the non-convexity of ACCQ. For convex regularizer, $\lambda_{\min}(\mathbf{x}_*) = O(\sqrt{s}\sigma)$ therefore is significantly sub-optimal compared to Eq. (12).

Remark 3 If the proximal projection gap $\alpha > 0$, Eq. (12) should be modified as :

$$\lambda_{\min}(\mathbf{x}_*) \geq O\left(\sqrt{\frac{\log d}{n}} + \alpha\right).$$

This is inferior than $\alpha = 0$. For $\alpha > 0$, the proof is similar. We will explore $\alpha > 0$ in the journal version of this paper.

3.4 PROOF SKETCH

In this analysis, we provide a sketch of proof for Theorem 1. The proof for Theorem 2 mostly follows the analysis of Theorem 1 by carefully exploiting the property of non-convex regularizer. All the detailed proofs can be found in the supplementary document.

The first step toward the proof for Theorem 1 is to show that the solution \mathbf{x}_{t+1} will be sparse if \mathbf{x}_t is sparse, which is revealed by the following theorem.

Theorem 3. Assume $|S_*| \leq s$, and $|S_t \setminus S_*| \leq s$. Then, if we set

$$\tau_t \geq \frac{3\delta}{\sqrt{s}} \|\mathbf{x}_t - \mathbf{x}_*\|_2 + 2\sigma \sqrt{\frac{\log d}{n}},$$

we have $|S_{t+1} \setminus S_*| \leq s$.

In the second step, we show in the theorem below that the recovery error will be reduced exponentially, up to the stochastic tolerance (i.e. $O(\sigma\sqrt{s \log d/n})$).

Theorem 4. Assume $|S_*| \leq s$, $|S_t \setminus S_*| \leq s$, and $\|\mathbf{x}_t - \mathbf{x}_*\|_2 \leq \Delta_t$. Then, by setting

$$\tau_t = \frac{3\delta}{\sqrt{s}} \Delta_t + 2\sigma \sqrt{\frac{\log d}{n}},$$

we have

$$\|\mathbf{x}_{t+1} - \mathbf{x}_*\|_2 \leq q\Delta_t + 4\sigma \sqrt{\frac{s \log d}{n}}$$

where

$$q = \max\left(3\delta, 2\sqrt{\frac{\delta}{1-2\delta}}\right)$$

With the above two theorems, we will show Theorem 1 by induction. Since $\|\mathbf{x}_1 - \mathbf{x}_*\|_2 = \|\mathbf{x}_*\|_2 \leq R$, Theorem 1 holds for $t = 1$. Let's assume that it holds for \mathbf{x}_t . Using Theorem 3, we have that \mathbf{x}_{t+1} is a $2s$ sparse vector with $|S_{t+1} \setminus S_*| \leq s$. Using Theorem 4, we have

$$\begin{aligned} \|\mathbf{x}_{t+1} - \mathbf{x}_*\|_2 &\leq q\Delta_t + 4\sigma \sqrt{\frac{s \log d}{n}} \\ &\leq q^t R + \frac{4\sigma}{1-q} \sqrt{\frac{s \log d}{n}} \end{aligned}$$

4 EXPERIMENTS

Since there are thousands of papers discussing compress sensing, it is impossible to compare every method published in this paper. We mainly select methods with theoretical guarantees and provable geometrical convergence rate. We compare Algorithm 1 (ACCQ) with five baseline methods:

- the greedy hard thresholding method (GraDeS) (Garg and Khandekar, 2009),
- Multiple Stage Capped ℓ_1 -norm method (MSCL1) (Zhang, 2012),
- non-convex proximal gradient descent with MCP (P-MCP) (Loh and Wainwright, 2013),
- the GIST method (Gong et al., 2013b), and
- Homotopy LASSO (LASSO) (Xiao and Zhang, 2012).

GraDeS is a greedy hard thresholding method with a geometrical convergence rate. We set $\gamma = 3$ in GraDeS as recommended in (Garg and Khandekar, 2009). Homotopy LASSO is a LASSO solver with geometrical convergence rate. We tune the regularizer parameter λ in Homotopy LASSO in set $\{1, 0.1, 10^{-2}, 10^{-3}, 10^{-4}\}$, and report the best performance. MSCL1, GIST and P-MCP are based on non-convex regularizers. The threshold parameter θ in MSCL1 is set to be $\theta = \lambda_{\min}(\mathbf{x}_*)/2 = 0.05$. Any $\theta < \lambda_{\min}(\mathbf{x}_*)$ should work as well (Zhang, 2012). Here we choose $\theta = \lambda_{\min}(\mathbf{x}_*)/2$ because it is at the same time large enough to suppress the noise. For the proposed algorithm, we similarly set $\theta = \lambda_{\min}(\mathbf{x}_*)/2 = 0.05$. We tune parameter q in the set $\{0.8, 0.9, 0.95, 0.99, 0.995\}$.

Two metrics are used to evaluate the recovery performance of different algorithms. To evaluate the recovery error, we follow compress sensing settings (Candès and Tao, 2005;

Table 2: Dataset Statistics

	d	$\ \mathbf{x}_*\ _0$	$\ \mathbf{x}_*\ _\infty$	$\ \mathbf{x}_*\ _2$	$\lambda_{\min}(\mathbf{x}_*)$
airplanes	$6.3 \times 10^4 \pm 6 \times 10^3$	$4.2 \times 10^2 \pm 92$	0.36 ± 0.059	0.96 ± 0.011	0.01
butterfly	$7 \times 10^4 \pm 1 \times 10^4$	$6.3 \times 10^2 \pm 2.6 \times 10^2$	0.28 ± 0.084	0.93 ± 0.045	0.01
camera	$7.4 \times 10^4 \pm 1.5 \times 10^4$	$5 \times 10^2 \pm 1.6 \times 10^2$	0.33 ± 0.11	0.94 ± 0.019	0.01
dolphin	$6.4 \times 10^4 \pm 1.1 \times 10^4$	$6 \times 10^2 \pm 2.1 \times 10^2$	0.27 ± 0.061	0.94 ± 0.027	0.01

Statistics of dataset of each category. d the dimension of \mathbf{x}_* . $\|\mathbf{x}_*\|_0$ is the number of non-zero entries in \mathbf{x}_* . $\|\mathbf{x}_*\|_\infty$ is the maximal amplitude of entries in \mathbf{x}_* . $\|\mathbf{x}_*\|_2$ is the ℓ_2 -norm of \mathbf{x}_* . $\lambda_{\min}(\mathbf{x}_*)$ is the smallest absolute value of the non-zero entries in \mathbf{x}_* . All numbers in the table are average values plus variances.

Table 3: Support Set Recovery Errors ϵ_{supp}

	LASSO	GraDeS	MSCL1	P-MCP	GIST	ACCQ
airplanes						
$\sigma = 0$	2.1×10^{-3}	0.0×10^0	0.0×10^0	2.0×10^{-1}	0.0×10^0	0.0×10^0
$\sigma = 0.01$	5.9×10^{-2}	8.2×10^{-3}	2.1×10^{-4}	2.0×10^{-1}	1.7×10^{-1}	0.0×10^0
$\sigma = 0.1$	6.0×10^{-2}	1.5×10^{-3}	5.3×10^{-4}	4.0×10^{-1}	4.4×10^{-2}	1.6×10^{-4}
butterfly						
$\sigma = 0$	3.8×10^{-2}	7.9×10^{-3}	4.4×10^{-3}	4.1×10^{-1}	5.9×10^{-3}	3.1×10^{-3}
$\sigma = 0.01$	3.9×10^{-2}	8.5×10^{-3}	4.7×10^{-3}	4.1×10^{-1}	1.0×10^{-1}	2.5×10^{-3}
$\sigma = 0.1$	6.3×10^{-2}	9.3×10^{-3}	1.1×10^{-2}	4.1×10^{-1}	8.4×10^{-3}	6.7×10^{-3}
camera						
$\sigma = 0$	5.2×10^{-3}	0.0×10^0	0.0×10^0	6.8×10^{-3}	0.0×10^0	0.0×10^0
$\sigma = 0.01$	5.4×10^{-2}	6.1×10^{-3}	5.0×10^{-3}	6.8×10^{-3}	9.7×10^{-2}	0.0×10^0
$\sigma = 0.1$	5.4×10^{-2}	2.6×10^{-3}	7.9×10^{-4}	6.8×10^{-3}	4.0×10^{-2}	1.5×10^{-4}
dolphin						
$\sigma = 0$	1.3×10^{-2}	3.8×10^{-3}	1.9×10^{-2}	2.1×10^{-1}	1.4×10^{-2}	0.0×10^0
$\sigma = 0.01$	2.3×10^{-2}	9.7×10^{-3}	1.9×10^{-2}	2.1×10^{-1}	1.4×10^{-2}	0.0×10^0
$\sigma = 0.1$	6.8×10^{-2}	5.9×10^{-3}	1.1×10^{-3}	2.1×10^{-1}	2.0×10^{-3}	1.5×10^{-4}

Support set recovery errors under different noise level. The smaller, the better.

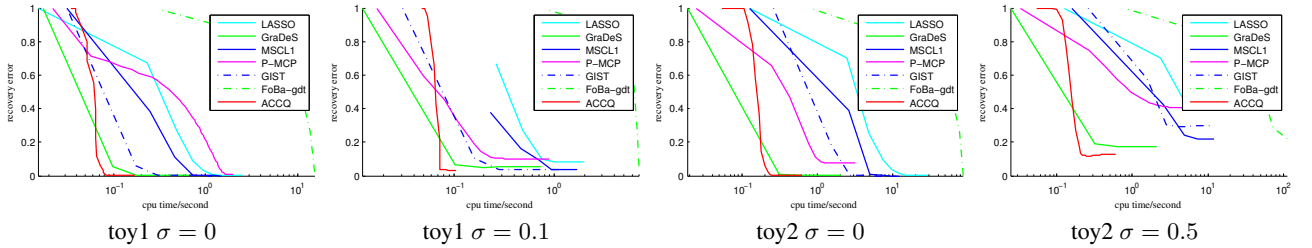


Figure 1: ℓ_2 Norm Recovery Errors For Synthetic Data Sets. x -Axis is CPU Time (Seconds) in Logarithmic Scale and y -Axis is ℓ_2 Norm Recovery Error $\|\mathbf{x}_t - \mathbf{x}_*\|$

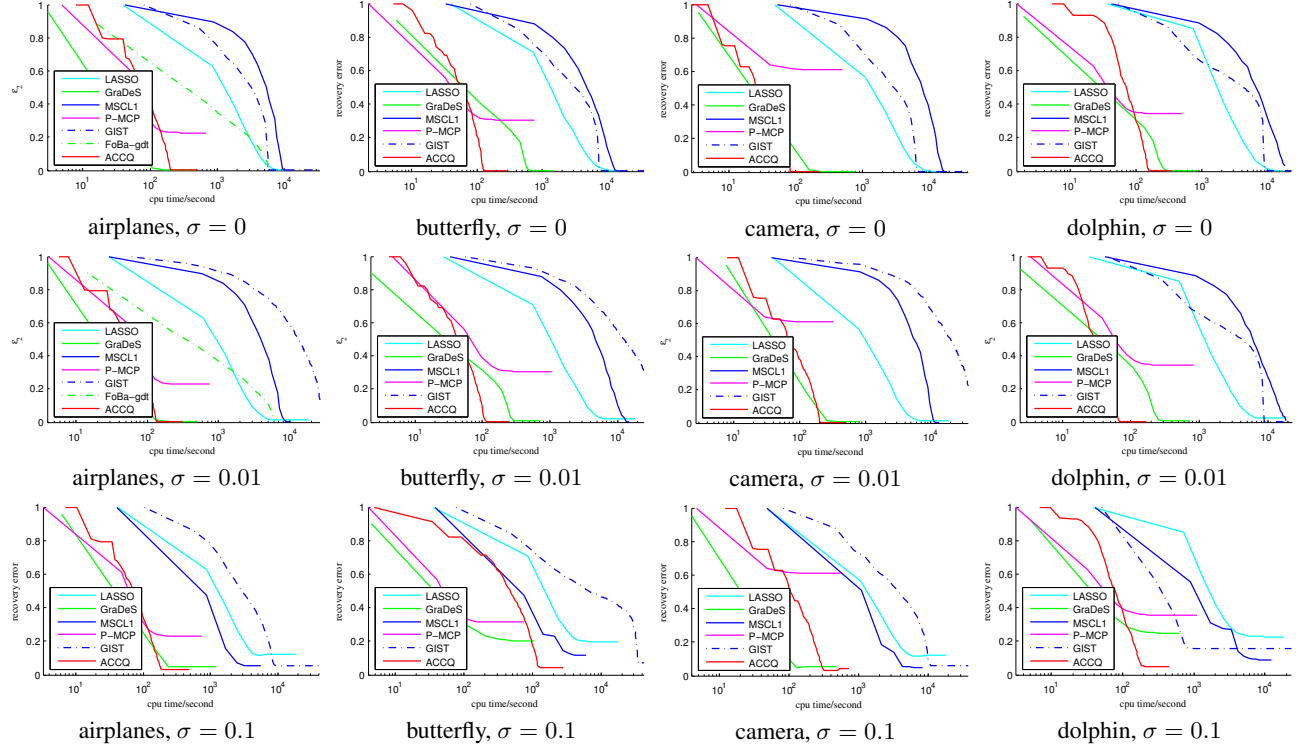


Figure 2: The Recovery Error For Real Images From The Caltech 101 Data Set

Zou and Hastie, 2005; Loh and Wainwright, 2012) and measure the ℓ_2 norm of the difference between the target vector and the recovered one. In order to take into the computational efficiency, we plot the recovery error vs. the running time for each algorithm. To evaluate the property of selection consistency, we also measure the accuracy in recovering the support set of the original sparse vector that defined as follows:

$$\epsilon_{\text{supp}} = (|S_t \setminus S_*| + |S_* \setminus S_t|) / d.$$

Therefore $\epsilon_{\text{supp}} = 0$ if and only if $S_t = S_*$.

All codes are implemented in Matlab, running on Intel(R) Core(TM)2 Duo CPU, P8700@2.53 GHz, Windows 7 64bit, 4GB memory. We terminate each algorithm if it runs more than five hours.

4.1 EXPERIMENTS WITH SYNTHETIC DATA

First we verify the effectiveness of the proposed algorithm on synthetic data. We generate A and \mathbf{z}_* independently from standard normal distributions. To generate the sparse vector \mathbf{x}_* , we first draw a random Gaussian vector \mathbf{x}'_* . We then normalize \mathbf{x}'_* to be one and only keep the largest s entries in \mathbf{x}'_* . We create two synthetic data sets: toy1 ($d = 1000$, $s = 50$, $\sigma = \{0, 0.1\}$, $n = 500$), and toy2 ($d = 5000$, $s = 50$, $\sigma = \{0, 0.5\}$, $n = 1000$). Clearly toy2 is more difficult than toy1 because of the high dimensionality and large noise. The performance averaged over 10 trials

is reported in this study. Since GraDeS needs to set all the entries in the intermediate solution to be zero except for the first k largest entries at each iteration, we tune k in set $\{40, 60, 80, 100\}$, and report the best performance.

Figure 1 shows the recovery results for the synthetic data sets, where the horizontal axis is CPU time (second) in the logarithmic scale. We observe that the proposed method ACCQ, although has the slow start at the beginning, is able to find a solution with small recovery error significantly faster than the other baseline methods. The slow start of the proposed algorithm is mostly due to the fact that the initial threshold τ_1 is set too high, leading to $\mathbf{x}_t = \mathbf{0}$ for the first a few of iterations. The LASSO bias is revealed by the noisy cases, where LASSO has the worst recovery error compared to the other methods. We also observe that the GraDeS method, an iterative hard thresholding algorithm, works well for the two toy datasets, in terms of both computational time and recovery error. We however found that for the real images, the GraDeS method behaves unstably when measurements are contaminated with random noise, as shown in the experimental result in the next subsection.

4.2 EXPERIMENTS WITH REAL IMAGES

Dataset We select a subset of images in Caltech101 as the sparse signals. Five images randomly chosen from four categories—“airplanes”, “butterfly”, “camera” and “dolphin”—are used in this study. For the convenience

of presentation, we denote each of the five images selected from one category by “001.jpg” to “005.jpg”. The total number of images we extract is 20. To generate a truly s -sparse signal, all images are first normalized to be zero-mean and unit variance. We then apply Fourier transform to the normalized image and filter out Fourier components with coefficient smaller than 0.01. The final sparse vector \mathbf{x}_* is constructed based on the survived Fourier components. Table 2 summarizes the statistics of our dataset.

For each s -sparse signal \mathbf{x}_* , we independently generate random Gaussian design matrix A with $n = 5000$. The entries in noise vector \mathbf{z} are independently drawn from a Gaussian distribution with variance σ varied in set $\{0, 0.01, 0.1\}$. When $\sigma > 0.1$, no algorithm could do a good job due to heavy information corruption. For parameter k in GraDeS, i.e. the number of non-zero entries to be kept at each iteration, we tune it in set $\{500, 1000, 2000\}$ and report the best performance.

Figure 2 presents the convergence rate of ℓ_2 -norm recovery error of the six methods. To save space we only show the result for “001.jpg” in each category here, and the results for the remaining images can be found in the supplementary document. Similar to the result of the synthetic data sets, The bias of LASSO is again revealed by its large recovery error for the noisy cases compared to several algorithms in comparison. The proposed algorithm ACCQ, although with a slow start, is able to find the solution with a small error significantly faster than the baseline algorithms on almost all cases except for airplanes with $\sigma = 0$ and camera with $\sigma = 0.1$, where the GraDeS method is the most efficient but with a slightly worse error. We notice that the performance of the GraDeS method appears to be not very consistent across images: it performs well for airplane and camera images, but does poorly for the images of butterfly and dolphin. Similarly, P-MCP and GIST, two sparse recovery algorithms with non-convex regularizers, although works well for the synthetic datasets, performs poorly for a number of cases. More investigation is needed to further understand the behavior of these algorithms.

In Table 2, we report the support set recovery error for each dataset under different noise level. Similar to ℓ_2 norm recovery error, ACCQ achieves perfect support set recovery on almost all datasets under $\sigma = \{0, 0.01\}$ except for *butterfly*. When noise $\sigma = 0.1$ is large, although ACCQ is unable to recover the exact support set, its error in recovering the support set is the smallest among the methods in comparison.

5 CONCLUSION

We propose an adaptive non-convex method to efficiently recover the sparse signal under compress sensing settings. The proposed method achieves a geometrical convergence rate for ℓ_2 -norm recovery error up to the statistical tol-

erance. By using a non-convex regularizer, the proposed method is able to remove the LASSO bias and achieve the selection consistency and oracle property. Experiments with both synthetic data sets and real images verify both the efficiency and effectiveness of the proposed method compared to the state-of-the-art methods for sparse recovery. In the future, we would like to improve our analysis to remove the extra condition in (13) for selection consistency and oracle property. We also plan to extend the proposed algorithm to other sparse recovery problems such as group sparsity and low rank matrix recovery.

Acknowledgments

The authors would like to thank the anonymous reviewers for their insightful suggestions. This work is supported by 973 Program(2013CB329503), NSFC (Grant No. 91120301) and Tsinghua National Laboratory for Information Science and TechnologyTNListCross-discipline Foundation.

References

- Alekh Agarwal, Sahand Negahban, and Martin J Wainwright. Fast global convergence of gradient methods for high-dimensional statistical recovery. *Ann. Stat.*, 40(5):2452–2482, 2012.
- Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm with application to wavelet-based image deblurring. In *ICASSP*, pages 693–696, 2009.
- Peter J Bickel, Ya’acov Ritov, and Alexandre B Tsybakov. Simultaneous analysis of lasso and dantzig selector. *Ann. Stat.*, 37(4):1705–1732, 2009.
- Emmanuel Candes and Terence Tao. The dantzig selector: statistical estimation when p is much larger than n . *Ann. Stat.*, pages 2313–2351, 2007.
- Emmanuel J Candes and Terence Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, 2005.
- Emmanuel J Candès, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.
- Emmanuel J Candes, Michael B Wakin, and Stephen P Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier Analysis and Applications*, 14(5-6):877–905, 2008.
- Mark A Davenport, Marco F Duarte, Yonina C Eldar, and Gitta Kutyniok. Introduction to compressed sensing. *Preprint*, 93, 2011.
- Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. *Ann. Stat.*, 32(2):407–499, 2004.
- J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1360, 2001.
- Simon Foucart. Sparse recovery algorithms: sufficient conditions in terms of restricted isometry constants. In *Approximation Theory XIII: San Antonio 2010*, pages 65–77. Springer, 2012.

- Simon Foucart and Ming-Jun Lai. Sparsest solutions of underdetermined linear systems via ℓ_q -minimization for $0 < q < 1$. *Applied and Computational Harmonic Analysis*, 26(3):395–407, 2009.
- Rahul Garg and Rohit Khandekar. Gradient descent with sparsification: an iterative algorithm for sparse recovery with restricted isometry property. In *ICML*, pages 337–344, 2009.
- Donald Geman and Chengda Yang. Nonlinear image recovery with half-quadratic regularization. *ITIP*, 4(7):932–946, 1995.
- Pinghua Gong, Jieping Ye, and Changshui Zhang. Multi-stage multi-task feature learning. *JMLR*, 14:2979–3010, 2013a.
- Pinghua Gong, Changshui Zhang, Zhaosong Lu, Jianhua Z. Huang, and Jieping Ye. A general iterative shrinkage and thresholding algorithm for non-convex regularized optimization problems. In *ICML*, 2013b.
- Jieping Ye Ji Liu, Ryohei Fujimaki. Forward-backward greedy algorithms for general convex smooth functions over a cardinality constraint. *arXiv:1401.0086*, 2013.
- V. Koltchinskii. *Oracle Inequalities in Empirical Risk Minimization and Sparse Recovery Problems*, volume 2033. Springer, 2011.
- Po-Ling Loh and Martin J Wainwright. High-dimensional regression with noisy and missing data: Provable guarantees with nonconvexity. *Ann. Stat.*, 40(3):1637–1664, 2012.
- Po-Ling Loh and Martin J Wainwright. Regularized m-estimators with nonconvexity: Statistical and algorithmic theory for local optima. *arXiv preprint arXiv:1305.2436*, 2013.
- Peter Ochs, Alexey Dosovitskiy, Thomas Brox, and Thomas Pock. An iterated ℓ_1 algorithm for non-smooth non-convex optimization in computer vision. In *CVPR*, 2013.
- Garvesh Raskutti, Martin J Wainwright, and Bin Yu. Minimax rates of convergence for high-dimensional regression under q -ball sparsity. In *Annual Allerton Conference on Communication, Control, and Computing*, pages 251–257, 2009.
- Mathieu Rosenbaum and Alexandre B Tsybakov. Sparse recovery under matrix uncertainty. *Ann. Stat.*, 38(5):2620–2651, 2010.
- R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.
- Joel A Tropp and Stephen J Wright. Computational methods for sparse solution of linear inverse problems. *Proceedings of the IEEE*, 98(6):948–958, 2010.
- Joshua Trzasko and Armando Manduca. Relaxed conditions for sparse signal recovery with general concave priors. *IEEE Transactions on Signal Processing*, 57(11):4347–4354, 2009.
- Sara A Van De Geer and Peter Bühlmann. On the conditions used to prove oracle results for the lasso. *Electronic Journal of Statistics*, 3:1360–1392, 2009.
- Shuo Xiang, Xiaotong Shen, and Jieping Ye. Efficient sparse group feature selection via nonconvex optimization. In *ICML*, 2013.
- Lin Xiao and Tong Zhang. A proximal-gradient homotopy method for the sparse least-squares problem. *arXiv preprint arXiv:1203.3002*, 2012.
- Huan Xu, Constantine Caramanis, and Shie Mannor. Robust regression and lasso. *IEEE Transactions on Information Theory*, 56(7):3561–3574, 2010.
- C.H. Zhang and T. Zhang. A general theory of concave regularization for high dimensional sparse estimation problems. *arXiv preprint arXiv:1108.4988*, 2011.
- Cun-Hui Zhang. Nearly unbiased variable selection under minimax concave penalty. *Ann. Stat.*, 38(2):894–942, 2010a.
- Tong Zhang. Analysis of multi-stage convex relaxation for sparse regularization. *JMLR*, 11:1081–1107, 2010b.
- Tong Zhang. Adaptive forward-backward greedy algorithm for learning sparse representation. *IEEE Transactions on Information Theory*, 57:4689–4708, 2011.
- Tong Zhang. Multi-stage convex relaxation for feature selection. *Bernoulli*, 2012.
- Tong Zhang Zhaoran Wang, Han Liu. Optimal computational and statistical rates of convergence for sparse nonconvex learning problems. *arXiv:1306.4960*, 2013.
- Hui Zou. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101(476):1418–1429, 2006.
- Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B Statistical Methodology*, 67(2):301–320, 2005.

Nuclear Norm Regularized Least Squares Optimization on Grassmannian Manifolds

Yuanyuan Liu
Hong Cheng

Dept. of Systems Engineering and Engineering
Management, The Chinese University of Hong Kong
{yyliu, hcheng}@se.cuhk.edu.hk

Fanhua Shang*
James Cheng

Dept. of Computer Science and Engineering
The Chinese University of Hong Kong
{fhshang, jcheng}@cse.cuhk.edu.hk

Abstract

This paper aims to address a class of nuclear norm regularized least square (NNLS) problems. By exploiting the underlying low-rank matrix manifold structure, the problem with nuclear norm regularization is cast to a Riemannian optimization problem over matrix manifolds. Compared with existing NNLS algorithms involving singular value decomposition (SVD) of large-scale matrices, our method achieves significant reduction in computational complexity. Moreover, the uniqueness of matrix factorization can be guaranteed by our Grassmannian manifold method. In our solution, we first introduce the bilateral factorization into the original NNLS problem and convert it into a Grassmannian optimization problem by using a linearized technique. Then the conjugate gradient procedure on the Grassmannian manifold is developed for our method with a guarantee of local convergence. Finally, our method can be extended to address the graph regularized problem. Experimental results verified both the efficiency and effectiveness of our method.

1 Introduction

In recent years, matrix approximation problems with nuclear norm regularization have occurred in many machine learning and compressed sensing applications such as matrix completion, matrix classification, multi-task learning and dimensionality reduction [6]. In this paper, we consider the following optimization problem over matrices:

$$\min_{X \in \mathbb{R}^{m \times n}} f(X) := g(X) + \mu \|X\|_*, \quad (1)$$

where $g(X)$ is any differentiable convex function (usually called the loss function, e.g. $g(X) = \|\mathcal{A}(X) - b\|_2^2$, where

$\mathcal{A}(\cdot)$ is a linear operator), $\mu > 0$ is a regularization parameter, and $\|X\|_*$ denotes the nuclear (or trace) norm of the matrix X with rank r , that is, the l_1 -norm of the matrix spectrum as $\|X\|_* = \sum_{i=1}^r \sigma_i$, where $\{\sigma_i\}$ are the singular values of X .

Most algorithms for solving the nuclear norm minimization (NNM) problem (1) do not require the rank to be specified and iteratively optimize the nuclear norm penalized problem. Naturally, the singular value decomposition (SVD) tends to play a critical computational role in the design of various nuclear norm solvers, e.g., the singular value thresholding (SVT) [5], soft-impute [14], accelerated proximal gradient approach [9], and so on. Those algorithms involving SVD and applying a soft-thresholding operator on the singular values at each iteration suffer from high computational cost of multiple SVDs [14, 15]. In particular, if the iterations need to pass through a region where the spectrum is dense, those algorithms can become potentially become prohibitively expensive [3]. Noticing that only those singular values exceeding a threshold and their corresponding singular vectors contribute to the soft-thresholding operator, a commonly used strategy is to compute the partial SVD instead of the full one, such as APGL [17] and IALM [12] both use PROPACK [11]. However, it can compute only a given number of largest singular values, and the soft-thresholding operator requires the principal singular values that are greater than a given threshold.

If the rank is known, a class of existing matrix factorization algorithms [10, 4, 22, 15, 18] cast the low-rank matrix estimation problem (1) as the following non-convex model,

$$\min_{X \in \mathbb{R}^{m \times n}} g(X), \quad \text{s.t., } \text{rank}(X) = k. \quad (2)$$

Matrix factorization is arguably the most widely applied method for the low-rank matrix completion problem, due to its high accuracy, scalability and flexibility to incorporating side-information [19]. LMaFit [22] fixes the rank by explicitly formulating the matrix as the product of its low-rank factors and using an optimization technique based on successive over-relaxation to solve (2). In [15] and [18], two improved versions were proposed to optimize it on the

*Corresponding author

Grassmannian manifolds, and improve its convergence by using conjugate gradients rather than the standard gradient descent. Moreover, [10] proved that exact recovery can be obtained with high probability by solving a non-convex optimization problem. In the model (2), the correct rank needs to be known as a priori. Unfortunately, the determination of the reduced rank is also an open problem, especially for the noisy matrix estimation.

To address these key problems mentioned above, we propose an effective approximation method for solving nuclear norm regularized least squares problems, which can reduce the SVD computational cost. We achieve it by converting the original NNM problem into a Grassmannian optimization problem. In our framework, we use the nuclear norm term to promote the robustness of the fixed-rank manifold optimization problem with respect to the given rank, in other words, to avoid the over-fitting problems of matrix factorization. Moreover, we present an efficient conjugate gradient descent algorithm on the Grassmannian manifolds with a guarantee of local convergence. Finally, our method is also extended to address the graph regularized problem. In summary, our method inherits the superiority of two classes of frameworks, i.e., the NNM methods and Riemannian manifold optimization methods based on matrix factorizations.

2 Background

When choosing $g(X) := \frac{1}{2} \|\mathcal{P}_\Omega(X) - \mathcal{P}_\Omega(Z)\|_F^2$ for some linear projection operators $\mathcal{P}_\Omega(\cdot)$, i.e., $\mathcal{P}_\Omega(X_{ij}) = X_{ij}$ if $(i, j) \in \Omega$, and $\mathcal{P}_\Omega(X_{ij}) = 0$ otherwise, the above formulation (1) is the low-rank matrix completion (MC) problem. The MC problem is to find out a matrix of the lowest rank whose entries in the observed set Ω correspond to the entries of Z :

$$\min_{X \in \mathbb{R}^{m \times n}} \frac{1}{2} \|\mathcal{P}_\Omega(X) - \mathcal{P}_\Omega(Z)\|_F^2 + \mu \|X\|_*, \quad (3)$$

or the noiseless version,

$$\min_{X \in \mathbb{R}^{m \times n}} \|X\|_*, \quad \text{s.t., } X_\Omega = Z_\Omega. \quad (4)$$

Recently, many low-complexity algorithms have emerged, such as APGL [17], IALM [12], and FPCA [13]. Involving SVDs in their each iteration, thus those algorithms based on the soft-thresholding operator suffer from high computational cost. This limits the usage of the matrix completion techniques in real-world applications.

Alternatively, low-rank matrix completion based on fixed-rank matrix factorization has received a significant amount of attention [15]. Suppose that the low-rank matrix $X \in \mathbb{R}^{m \times n}$ with rank r is decomposed as $X = UM$, where $U \in \mathbb{R}^{m \times r}$ and $M \in \mathbb{R}^{r \times n}$. LMaFit [22] applies a successive over-relaxation iteration scheme to alternatively solve the

following least-squares problem,

$$\min_{U \in \mathbb{R}^{m \times r}} \min_{M \in \mathbb{R}^{r \times n}} \sum_{(i,j) \in \Omega} \|(UM)_{ij} - Z_{ij}\|^2. \quad (5)$$

However, the factorization of the matrix X into the product UM is not unique. Indeed, for any r -by- r invertible matrix O , we have $UM = (UO)(O^{-1}M)$. Hence, some researchers convert the matrix factorization problem into some corresponding Riemannian manifold optimization problems, such as OptSpace [10], RTRMC [4], RiemannCG [18], and ScGrass [15]. However, in those algorithms we need to know the exact rank which is usually difficult to obtain. Furthermore, they often suffer badly from overfitting due to their least-squares loss functions, especially on noisy matrices.

2.1 Grassmannian Manifold

We will briefly recall the related notions of matrix manifolds (readers may refer to [2] for details).

Definition 1. Grassmannian manifold: The set of r -dimensional vector subspaces of \mathbb{R}^m is defined as $\mathcal{G}_{m,r}$. Each point $\mathcal{U} \in \mathcal{G}_{m,r}$ can be presented by a generator matrix $U \in \mathcal{N}_{m,r}$, where $\mathcal{N}_{m,r}$ is the set of $m \times r$ matrices with orthonormal columns, i.e., $\mathcal{N}_{m,r} = \{U \in \mathbb{R}^{m \times r} : U^T U = I_r\}$.

Definition 2. Tangent space: Consider an arbitrary point on the Grassmannian manifold, $\mathcal{U} \in \mathcal{G}_{m,r}$. To perform differential calculus, the tangent space at U (the generator matrix of \mathcal{U}) is denoted as $T_U \mathcal{G}_{m,r}$. And the tangent space is represented as $T_U \mathcal{G}_{m,r} = \{\eta \in \mathbb{R}^{m \times r} : U^T \eta = 0\}$.

As the generalization of the standard optimization methods, some Riemannian manifold optimization methods can be used for solving the following low-rank matrix learning problem with the fixed rank r ,

$$\min_{U \in \mathcal{G}_{m,r}} f(U), \quad (6)$$

where $f(\cdot)$ is a smooth function on Grassmannian manifolds.

2.2 Skeleton of CG Algorithms on Grassmannian Manifolds

In general, the typical nonlinear conjugate gradient (CG) algorithm on Grassmannian manifolds with a line-search rule for the unconstrained optimization problem (6) is outlined in **Algorithm 1**, which we elaborate as follows.

- Ambient gradient: To obtain the Euclidean gradient $\nabla f(U_k)$ in the ambient space.
- Riemannian gradient: It, denoted by $\text{grad} f(U_k)$, is a specific tangent vector η_k which corresponds to the

Algorithm 1 Geometric CG

Input: The fixed rank r and $\text{tol} > 0$.**Output:** $X = UM$.

- 1: **while** not converged **do**
 - 2: Compute the ambient gradient: $\nabla f(U_k)$.
 - 3: Compute the Grassmannian gradient:
 $\text{grad}f(U_k)$.
 - 4: Check convergence: $\|\text{grad}f(U_k)\| \leq \text{tol}$.
 - 5: Compute β_k by the PR+ updating rule,
 and compute a conjugate direction ξ_k :
 $\xi_k = -\text{grad}f(U_k) + \beta_k T_{U_{k-1} \rightarrow U_k} \mathcal{G}_{m,r}(\xi_{k-1})$.
 - 6: Find an appropriate step size t_k and compute
 $U: U_{k+1} = \mathcal{R}_{U_k}(t_k \xi_k)$.
 - 7: **end while**
-

direction of steepest ascent of $f(U_k)$, but is restricted to only directions in the tangent space $T_{U_k} \mathcal{G}_{m,r}$.

- The conjugate direction: It, denoted by $\xi_k \in T_{U_k} \mathcal{G}_{m,r}$, is conjugate to the gradient, and requires taking a linear combination of the Riemannian gradient with the previous search direction ξ_{k-1} . Since ξ_{k-1} does not lie in $T_{U_k} \mathcal{G}_{m,r}$, it needs to be transported to $T_{U_k} \mathcal{G}_{m,r}$. This is done by a mapping $T_{U_{k-1} \rightarrow U_k} \mathcal{G}_{m,r} : T_{U_{k-1}} \mathcal{G}_{m,r} \rightarrow T_{U_k} \mathcal{G}_{m,r}$, the so-called *vector transport*. In total, the conjugate direction $\xi_k = -\text{grad}f(U_k) + \beta_k T_{U_{k-1} \rightarrow U_k} \mathcal{G}_{m,r}(\xi_{k-1})$ can be computed by a variant of the classical Polak-Ribière (PR+) updating rule in the non-linear CG.
- Retraction: Because a tangent vector only gives a direction but not the line search itself on the manifold, a smooth mapping $\mathcal{R}_{U_k} : T_{U_k} \mathcal{G}_{m,r} \rightarrow \mathcal{G}_{m,r}$, named as *retraction*, is needed to map tangent vectors to the manifold. To retract the search direction ξ_k with a line-search step size t_k back to the manifold is denoted as: $U_{k+1} = \mathcal{R}_{U_k}(t_k \xi_k)$.

3 Grassmannian Optimization

3.1 Linearization Technique

As in [17], the problem (1) can be approximated iteratively by minimizing the following linearized function,

$$\begin{aligned} \mathcal{L}(X) = & \mu \|X\|_* + g(X_k) + \langle \nabla g(X_k), X - X_k \rangle \\ & + \frac{1}{2\tau} \|X - X_k\|_F^2, \end{aligned} \quad (7)$$

where $\tau > 0$ is a proximal parameter. Without loss of generality, suppose d is an upper bound for $\text{rank}(X) = r$, i.e., $r \leq d$. $X \in \mathbb{R}^{m \times n}$ is decomposed as $X = UM$, where $U \in \mathcal{N}_{m,d}$ and $M \in \mathbb{R}^{d \times n}$. Furthermore, the quotient geometry (i.e., Grassmannian manifold) is used in our paper to guarantee the uniqueness of matrix factorization.

Hence, $U \in \mathcal{N}_{m,d}$ can be viewed as the generator matrix of $\mathcal{U} \in \mathcal{G}_{m,d}$ and is an orthonormal basis of \mathcal{U} . With $U^T U = I$, we have $\|X\|_* = \|M\|_*$. Thus, the problem (7) is rewritten in the following form

$$\begin{aligned} \mathcal{L}(U, M) = & \mu \|M\|_* + \langle \nabla g(U_k M_k), UM - U_k M_k \rangle \\ & + g(U_k M_k) + \frac{1}{2\tau} \|UM - U_k M_k\|_F^2. \end{aligned} \quad (8)$$

For solving the problem (8), then we formulate the following subproblem at the k -th iteration,

$$\begin{aligned} \min_{U \in \mathcal{G}_{m,d}} \min_{M \in \mathbb{R}^{d \times n}} \tilde{f}_{U_k M_k}(U, M) := \\ \mu \tau \|M\|_* + \frac{1}{2} \|UM - U_k M_k + \tau \nabla g(U_k M_k)\|_F^2. \end{aligned} \quad (9)$$

In the following, the problem (9) is equally converted into a Grassmannian manifold optimization problem with respect to U .

3.2 Objective Function on Grassmannian Manifolds

Similar to [4], we now derive the objective function on Grassmannian manifolds. Given the variable U , computing the matrix M that minimizes $\tilde{f}_{U_k M_k}$ is a nuclear norm regularized least-squares problem. The mapping between U and this (unique) optimal M , denoted by M_U , is given by

$$\begin{aligned} U \mapsto M_U = \\ \arg \min_{M \in \mathbb{R}^{d \times n}} \mu \tau \|M\|_* + \frac{1}{2} \|UM - P_k\|_F^2, \end{aligned} \quad (10)$$

where $P_k = U_k M_k - \tau \nabla g(U_k M_k)$. Following [5], we can obtain a unique closed-form solution to the problem (10) via the SVT operator,

$$M_U = \text{SVT}_{\mu\tau}(U^T P_k), \quad (11)$$

where $\text{SVT}_{\mu\tau}(A) := \bar{U} \text{diag}(\max\{\sigma - \mu\tau, 0\}) \bar{V}$ and $\bar{U} \text{diag}(\sigma) \bar{V}$ is the SVD of A . Substituting (11) into the function $\tilde{f}_{U_k M_k}$, then the cost function $f_{U_k M_k} : \mathcal{G}_{m,r} \rightarrow \mathbb{R}$ on Grassmannian manifolds is given by

$$\min_{U \in \mathcal{G}_{m,d}} f_{U_k M_k}(U) := \mu \tau \|M_U\|_* + \frac{1}{2} \|UM_U - P_k\|_F^2. \quad (12)$$

3.3 Riemannian Gradient

For solving our problem (12), we first derive the formulas for the Euclidean gradient of the cost function $f_{U_k M_k}$ in (12) at U . Using the chain rule, we have

$$\begin{aligned} \nabla f_{U_k M_k}(U) &= \frac{d}{dU} f_{U_k M_k}(U) \\ &= \frac{\partial}{\partial U} \tilde{f}_{U_k M_k}(U, M_U) + \frac{\partial}{\partial M_U} \tilde{f}_{U_k M_k}(U, M_U) \frac{d}{dU} M_U, \end{aligned} \quad (13)$$

where $\tilde{f}_{U_k M_k}(U, M_U)$, $f_{U_k M_k}(U)$ and the map M_U have been defined in (9), (12) and (11), respectively. The first term of (13), $\frac{\partial}{\partial U} \tilde{f}_{U_k M_k}(U, M_U)$, can be computed easily. To compute the second term in (13), i.e., $\frac{\partial}{\partial M_U} \tilde{f}_{U_k M_k}(U, M_U) \frac{d}{dU} M_U$, we will present the following derivation using the singular value and singular subspace perturbation theories.

3.3.1 Computation of Ambient Gradient

To compute the ambient gradient, we first introduce the following two definitions and give their property, respectively.

Definition 3. Subdifferential: Let $\partial_M \tilde{f}_{U_k M_k}(U, M)$ denote the subdifferential of the non-smooth function $\tilde{f}_{U_k M_k}(U, M)$ at M , then

$$\partial_M \tilde{f}_{U_k M_k}(U, M) = \mu\tau \partial \|M\|_* + (M - U^T P_k), \quad (14)$$

where $\partial \|\cdot\|_*$ denotes the subdifferential of the non-smooth convex function $\|\cdot\|_*$, and is a closed convex set. Specifically, let $M = \hat{U} \hat{\Sigma} \hat{V}$ be the SVD of $M \in \mathbb{R}^{d \times n}$, then $\partial \|M\|_*$ is given by [5], i.e.,

$$\begin{aligned} \partial \|M\|_* = \\ \{\hat{U} \hat{V} + W : \hat{U}^T W = 0, W \hat{V}^T = 0, \|W\|_2 \leq 1\}, \end{aligned} \quad (15)$$

where $\|\cdot\|_2$ is a spectrum norm.

By Definition 3, we can obtain the following property.

Lemma 1. Let M_U be the solution of problem (10), $M_U = \tilde{U} \tilde{\Lambda} \tilde{V}$ be the SVD of M_U , and $\Gamma = \{W : \tilde{U}^T W = 0, W \tilde{V} = 0, \|W\|_2 \leq 1\}$, then $\exists \tilde{W} \in \Gamma$ satisfies

$$\partial_{M_U} \tilde{f}_{U_k M_k} = \{\mu\tau(W - \tilde{W}), W \in \Gamma\}. \quad (16)$$

Proof. Since M_U is a optimal solution, then the first-order optimality condition of the problem (10) is given by,

$$0 \in \mu\tau \partial \|M_U\|_* + (M_U - U^T P_k). \quad (17)$$

By (15), then $\exists \tilde{W} \in \Gamma$ satisfies

$$\mu\tau \tilde{U} \tilde{V} + \mu\tau \tilde{W} + (M_U - U^T P_k) = 0. \quad (18)$$

Furthermore, substituting (18) into the subdifferential in (14), we have

$$\begin{aligned} \partial_M \tilde{f}_{U_k M_k} &= \mu\tau \partial \|M_U\|_* + (M_U - U^T P_k) \\ &= \{\mu\tau \tilde{U} \tilde{V}^T + \mu\tau W + M - U^T P_k, W \in \Gamma\} \\ &= \{\mu\tau(W - \tilde{W}), W \in \Gamma\}. \end{aligned} \quad (19)$$

This completes the proof. \square

Definition 4. Directional Derivative: Let $M_U = \text{SVT}_{\mu\tau}(U^T P_k)$, the directional derivative of the mapping M_U at U along the direction H is defined as

$$M_{U,H} = \lim_{\gamma \rightarrow 0} \frac{\text{SVT}_{\mu\tau}((U + \gamma H)^T P_k) - \text{SVT}_{\mu\tau}(U^T P_k)}{\gamma}. \quad (20)$$

Furthermore, we give the following result by the singular value and singular subspaces perturbation theorems.

Lemma 2. With the same notations as Lemma 1, then for any matrix $W \in \Gamma$, we have

$$\langle M_{U,H}, W \rangle = 0. \quad (21)$$

Proof. To prove the lemma, the classical perturbation theory for singular value and singular subspaces problems is introduced. We use the classical result of [20] that the eigenvalues of a matrix which is an analytic function of a single variable can always be numbered so that they are each analytic functions of the variable. Using the relationship between eigenvalues and singular values, it follows that if the singular values of the matrix $B = A + \gamma R$, where A and R are $m \times n$ matrices, denoted by $\sigma_i(\gamma)$, $i = 1, 2, \dots, n$, then

$$\sigma_i(\gamma) = \sigma_i + \gamma u_i^T R v_i + O_i(\gamma), i = 1, 2, \dots, n, \quad (22)$$

where $O_i(\gamma)$ is an infinitesimal of higher order than γ , u_i and v_i are singular vectors of A corresponding to σ_i . Let $A = \tilde{U} \tilde{\Sigma} \tilde{V}$ and $B = \hat{U} \hat{\Sigma} \hat{V}$ be the SVDs of A and B respectively, where $\tilde{\Sigma} = \text{diag}(\sigma_1, \dots, \sigma_s)$ and $\hat{\Sigma} = \text{diag}(\sigma_1(\gamma), \dots, \sigma_s(\gamma))$ are s largest singular values of A and B , respectively. \tilde{U} and \hat{U} denote the spaces of \tilde{U} and \hat{U} , and \tilde{V} and \hat{V} denote the spaces of \tilde{V} and \hat{V} , respectively. Then the classic theorem on the perturbation of singular subspaces is due to [21],

$$\begin{aligned} &\sqrt{\|\sin \Theta(\tilde{U}, \hat{U})\|_F^2 + \|\sin \Theta(\tilde{V}, \hat{V})\|_F^2} \\ &\leq \frac{\sqrt{\|E_1\|_F^2 + \|E_2\|_F^2}}{\delta}, \end{aligned} \quad (23)$$

where $E_1 = B \tilde{V} - \tilde{U} \tilde{\Sigma} \equiv \gamma R \tilde{V}$, $E_2 = B^T \tilde{U} - \tilde{V} \tilde{\Sigma} \equiv \gamma R^T \tilde{U}$, and $\|\sin \Theta(\tilde{U}, \hat{U})\|_F^2$ is a measure that is related to the canonical angles between the subspace \tilde{U} and \hat{U} . Moreover, the gap δ is the distance between two sets of singular values in $\tilde{\Sigma} = \text{diag}(\sigma_1, \dots, \sigma_s)$ and $\tilde{\Sigma}' = \text{diag}(\sigma_{s+1}, \dots, \sigma_n)$ (Please see the details in [21]).

Let $A := U^T P_k$, $R := H^T P_k$, $B := A + \gamma R = U^T P_k + \gamma H^T P_k$, $\sigma_1 \geq \dots \geq \sigma_s \geq \mu\tau$ and $\sigma_{s+1} < \mu\tau$. By using the result in (22) and the definition of the SVT operator with $\gamma \rightarrow 0$, we have

$$\begin{aligned} &\text{SVT}_{\mu\tau}((U + \gamma H)^T P_k) - \text{SVT}_{\mu\tau}(U^T P_k) \\ &= \hat{U}(\hat{\Sigma} - \mu\tau)\hat{V} - \tilde{U}(\tilde{\Sigma} - \mu\tau)\tilde{V} \\ &= T_1 + T_2 + O(\gamma), \end{aligned} \quad (24)$$

where $T_1 = \hat{U}(\tilde{\Sigma} - \mu\tau)\hat{V} - \tilde{U}(\tilde{\Sigma} - \mu\tau)\tilde{V}$, $T_2 = \gamma \hat{U} \Delta \hat{V}$, the i -th element of the diagonal matrix Δ is $\Delta_i = \tilde{u}_i^T H^T P_k \tilde{v}_i$, $O(\gamma) \in \mathbb{R}^{d \times n}$ and its all entries are infinitesimals of higher order than γ .

By the singular subspace perturbation theory of in (23), the subspace $\hat{U} \rightarrow \tilde{U}$, while $\gamma \rightarrow 0$, i.e., $\exists D_1$, such that $\hat{U} \rightarrow$

$\tilde{U}D_1$. Similarly, $\exists D_2$, such that $\hat{V} \rightarrow \tilde{V}D_2$. Thus, it is not difficult to verify $\hat{U} = \tilde{U}D_1 + \delta_1(\gamma)$ and $\hat{V}^T = \tilde{V}^TD_2 + \delta_2(\gamma)^T$, where $\delta_1(\gamma) \in \mathbb{R}^{m \times d}$ and $\delta_2(\gamma) \in \mathbb{R}^{d \times n}$, and all of their entries are infinitesimals of the same order as γ . Then we have

$$\begin{aligned} \langle W, \hat{U}(\tilde{\Sigma} - \mu\tau)\hat{V} \rangle &= \langle \hat{U}^TW\hat{V}^T, (\tilde{\Sigma} - \mu\tau) \rangle \\ &= \langle D_1^T\tilde{U}^TW\tilde{V}^T, (\tilde{\Sigma} - \mu\tau) \rangle \\ &\quad + \langle \delta_1(\gamma)^TW\tilde{V}^TD_2, (\tilde{\Sigma} - \mu\tau) \rangle \\ &\quad + \langle \delta_1(\gamma)^TW\delta_2(\gamma), (\tilde{\Sigma} - \mu\tau) \rangle, \end{aligned} \quad (25)$$

where W is defined in (15), $\tilde{U}^TW = 0$, $W\tilde{V}^T = 0$, and by (25), then

$$\begin{aligned} \langle W, T_1 \rangle &= \langle W, \hat{U}(\tilde{\Sigma} - \mu\tau)\hat{V} \rangle + \langle W, \tilde{U}(\tilde{\Sigma} - \mu\tau)\tilde{V} \rangle \\ &= \langle \delta_1(\gamma)^TW\delta_2(\gamma), (\tilde{\Sigma} - \mu\tau) \rangle. \end{aligned}$$

Similarly, we have

$$\begin{aligned} \langle W, T_2 \rangle &= \langle W, \gamma\hat{U}\Delta\hat{V} \rangle \\ &= \langle D_1^T\tilde{U}^TW\hat{V}^T, \gamma\Delta \rangle + \langle \delta_1^TW\tilde{V}^TD_2, \gamma\Delta \rangle \\ &\quad + \langle \delta_1^T(\gamma)W\delta_2(\gamma), \gamma\Delta \rangle \\ &= \langle \delta_1^T(\gamma)W\delta_2(\gamma), \gamma\Delta \rangle. \end{aligned}$$

Thus, we have

$$\langle W, M_{U,H} \rangle = \lim_{\gamma \rightarrow 0} \frac{\langle W, T_1 + T_2 + O(\gamma) \rangle}{\gamma} = 0.$$

Thus, this completes the proof. \square

Next we will compute the ambient gradient. Let $\forall \zeta \in \frac{\partial}{\partial M_U} \tilde{f}_{U_k M_k}(U, M_U) \frac{d}{dU} M_U$, by the chain rule of composite function, and substituting the result in Lemma 1 into the chain rule, then $\exists (W - \tilde{W}) \in \Gamma$ satisfies

$$\begin{aligned} \zeta_{ij} &= \langle W - \tilde{W}, M_{U, \tilde{H}^{ij}} \rangle, \\ i &= 1, 2, \dots, m, \quad j = 1, 2, \dots, d, \end{aligned}$$

where ζ_{ij} denotes the element in the i -th row and the j -th column of ζ , and $M_{U, \tilde{H}^{ij}}$ is given by Definition 2, and the direction $\tilde{H}^{ij} \in \mathbb{R}^{m \times d}$ is defined as

$$(\tilde{H}^{ij})_{m,n} = \begin{cases} 1 & m = i \text{ and } n = j, \\ 0 & \text{otherwise.} \end{cases}$$

And by Lemma 2, then

$$\zeta_{ij} = \langle W - \tilde{W}, M_{U, \tilde{H}^{ij}} \rangle = 0. \quad (26)$$

Thus, we have

$$\frac{\partial}{\partial M_U} \tilde{f}_{U_k M_k}(U, M_U) \frac{d}{dU} M_U = 0. \quad (27)$$

By (27), then the ambient gradient in (13) can be rewritten as follows:

$$\begin{aligned} \nabla f_{U_k M_k}(U) &= \frac{\partial}{\partial U} \tilde{f}_{U_k M_k}(U, M_U) \\ &= (UM_U - P_k)M_U^T. \end{aligned} \quad (28)$$

Note the above result implies that the function $f_{U_k M_k}(\cdot)$ is continuously differentiable.

3.3.2 Computation of Riemannian Gradient

Following [18], and $(I - UU^T)U = 0$, then the Grassmannian gradient of $f_{U_k M_k}$ at U is given by

$$\begin{aligned} \text{grad} f_{U_k M_k}(U) &= (I - UU^T) \nabla f_{U_k M_k}(U) \\ &= -(I - UU^T)P_k M_U^T. \end{aligned} \quad (29)$$

3.4 Conjugate Gradient Iteration

In the part, we describe the nonlinear CG algorithm on the Grassmannian manifold for solving the proposed model. The main additional ingredient we need is vector transport which is used to transport the old search direction to the current point on the manifold, i.e., $T_{U_{k-1} \rightarrow U_k} \mathcal{G}_{m,d} : T_{U_{k-1}} \mathcal{G}_{m,d} \rightarrow T_{U_k} \mathcal{G}_{m,d}$. The transport search direction is then combined with the gradient at the current point, e.g. by the Polak-Ribière formula (see [2]), to derive the new search direction. Vector transport can be defined using the Riemann connection, which in turn is defined based on the Riemann metric [1]. In this paper, we will use the canonical metric to derive vector transport when considering the natural quotient manifold structure of the Grassmannian manifold. Following [15], the previous search direction ξ_{k-1} at U_{k-1} will be transported to U_{k+1} as $T_{U_{k-1} \rightarrow U_k} \mathcal{G}_{m,d} = (I - U_k U_k^T) \xi_{k-1}$. Then the new search direction is

$$\xi_k = -\text{grad} f(U_k) + \beta_k T_{U_{k-1} \rightarrow U_k} \mathcal{G}_{m,r}(\xi_{k-1}), \quad (30)$$

where β_k can be calculated by using the Polak-Ribiere formula in [7].

Furthermore, U is updated by

$$U_{k+1} = R(U_k + t_k \xi_k) = \text{qf}(U_k + t_k \xi_k), \quad (31)$$

where $\text{qf}(A)$ is used as a retraction operator, which is the Q factor in the QR factorization of A , and the step size t_k is obtained by the Armijo linear search rule [2].

To solve the Riemannian optimization subproblem (12) at each iteration, we present a non-linear conjugate gradient decent algorithm on Grassmannian manifolds. Overall, the skeleton of our method is listed in **Algorithm 2**.

3.5 Convergence Analysis

In this part, we analyze the convergence of Algorithm 2 using the non-linear conjugate gradient descent scheme.

Algorithm 2 A Riemannian optimization framework for solving the problem (12)

Input: The rank d , the parameters μ, τ and tol .

Output: $X = UM$.

- 1: **while** not converged **do**
- 2: Formulate the cost function f_{UM_k} by (12).
- 3: Compute the Grassmannian gradient by (29),
 $\eta_k = \text{grad} f_{UM_k}(U_k)$.
- 4: Check convergence: $\eta_k \leq \text{tol}$.
- 5: Compute a conjugate direction ξ_k by (30).
- 6: Find an appropriate step size t_k using Armijo rule,
and compute U_{k+1} by (31).
- 7: Compute M_{k+1} by (11).
- 8: **end while**

Lemma 3. Let $g(X) = \|A(X) - b\|_F^2$, $\{(U_k, M_k)\}$ be an infinite sequence of iterates generated by Algorithm 2 with the Armijo backtracking rule, and $\tau \in (0, 1/\rho(\mathcal{A}^T \mathcal{A}))$, where $\rho(\cdot)$ denotes the spectral radius operator, then we have the following results:

(I) $\lim_{k \rightarrow \infty} \|\text{grad} f_{UM_k}(U_k)\| = 0$.

(II) $\lim_{k \rightarrow \infty} \|U_{k+1} - U_k\| = 0$, and

$$\lim_{k \rightarrow \infty} \|U_{k+1}M_{k+1} - U_kM_k\| = 0.$$

Proof: The detailed proof can be found in the supplementary material.

Theorem 4. Let $\{(U_k, M_k)\}$ be an infinite sequence of iterates generated by Algorithm 2. Then each accumulation point of $\{(U_k, M_k)\}$ is a critical point of the following optimization problem

$$\min_{\mathcal{U} \in \mathcal{G}_{m,d}} \min_{M \in \mathbb{R}^{d \times n}} \mu \tau \|M\|_* + \frac{1}{2} g(UM). \quad (32)$$

Proof: The detailed proof of the theorem can be found in the supplementary material.

3.6 Complexity Analysis

In this part, we discuss the time complexity of our method. For the matrix completion problem (12), the main running time of our algorithm is consumed by performing SVD for the SVT operator, some multiplications and retraction operator. The time complexity of performing the SVT operator in (11) is $O_1 := O(d^2n)$. The time complexity of some multiplication and retraction operators is $O_2 := O(dmn + d^2m)$. The time complexity of performing retraction operator is $O_3 := O(d^2m)$. Thus, the total time complexity of our method is $O(T(O_1 + O_2 + O_3))$, where T is the number of iterations.

4 Graph Regularization Extensions

In this paper, we mainly consider the problem of recovering a noisy low-rank matrix from a few observed entries as

a matrix completion application of our nuclear norm regularized least squares model. In addition, our method is quite general, and can be easily extended to incorporate the contextual information, including social relations of users, social tags issued by users, movie genres, user demographic information, etc. In order to incorporate the social network information, our social network aided context-aware recommender model is formulated as follows:

$$\min_{U \in \mathcal{N}_{m,d}} \min_{M \in \mathbb{R}^{d \times n}} f(U, M) := \frac{1}{2} \|\mathcal{P}_\Omega(UM) - \mathcal{P}_\Omega(Z)\|_F^2 + \mu \|M\|_* + \frac{\lambda_1}{2} \text{tr}(U^T L_U U) + \frac{\lambda_2}{2} \text{tr}(M L_M M^T), \quad (33)$$

where $\text{tr}(A)$ denotes the trace of the matrix A , L_U and L_M are the graph Laplacian matrices, i.e., $L_U = D_U - W_U$, W_U is the weight matrix for the user set, and D_U is the diagonal matrix whose entries are column sums of W_U , i.e., $(D_U)_{ii} = \sum_j (W_U)_{ij}$, and $\lambda_1 \geq 0$ and $\lambda_2 \geq 0$ are regularization constants. It is not difficult to verify, for any matrix $O \in \mathcal{N}_{d,d}$, we have $f(UO, O^T M) = f(U, M)$. Hence, the Grassmannian manifold is also used in our social network aided context-aware recommender model, and it is reformulated as follows:

$$\min_{\mathcal{U} \in \mathcal{G}_{m,d}} \min_{M \in \mathbb{R}^{d \times n}} \frac{1}{2} \|\mathcal{P}_\Omega(UM) - \mathcal{P}_\Omega(Z)\|_F^2 + \mu \|M\|_* + \frac{\lambda_1}{2} \text{tr}(U^T L_U U) + \frac{\lambda_2}{2} \text{tr}(M L_M M^T), \quad (34)$$

where the column orthonormal matrix U is viewed as the generator matrix of \mathcal{U} . Moreover, Algorithm 2 can be extended to solve our graph regularized matrix completion problem (34).

5 Experimental Results

In this section, we evaluate both the effectiveness and efficiency of our method for solving matrix completion problems on both synthetic and real-world data.

5.1 Synthetic Data

The synthetic matrices $Z \in \mathbb{R}^{m \times n}$ with rank r in this subsection are created randomly by the following procedure: two random matrices $U \in \mathbb{R}^{m \times r}$ and $V \in \mathbb{R}^{n \times r}$ with i.i.d. standard Gaussian entries are first generated, and then $X = UV^T$ is assembled. Two test experiments are conducted on random matrix without or with noise, where the observed subset is corrupted by i.i.d. standard Gaussian random variables as in [17]. In both cases, only 10% observed entries are sampled uniformly at random as the training set, and the remaining is used as the testing set. Summaries of the computational results are presented in Figure 1 on noiseless matrices of size 2000×2000 and

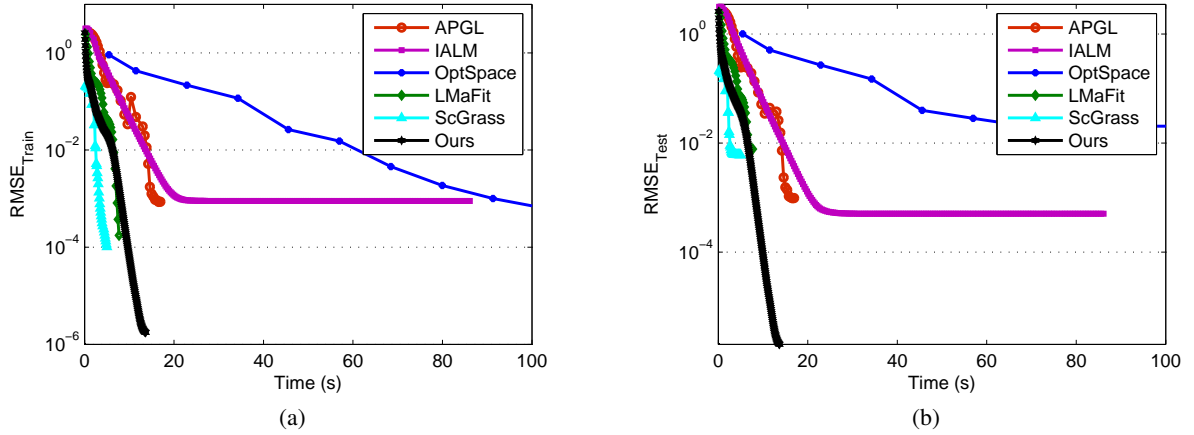


Figure 1: The recovery accuracy on noiseless data vs. running time (seconds): training RMSE (left) and testing RMSE (right).

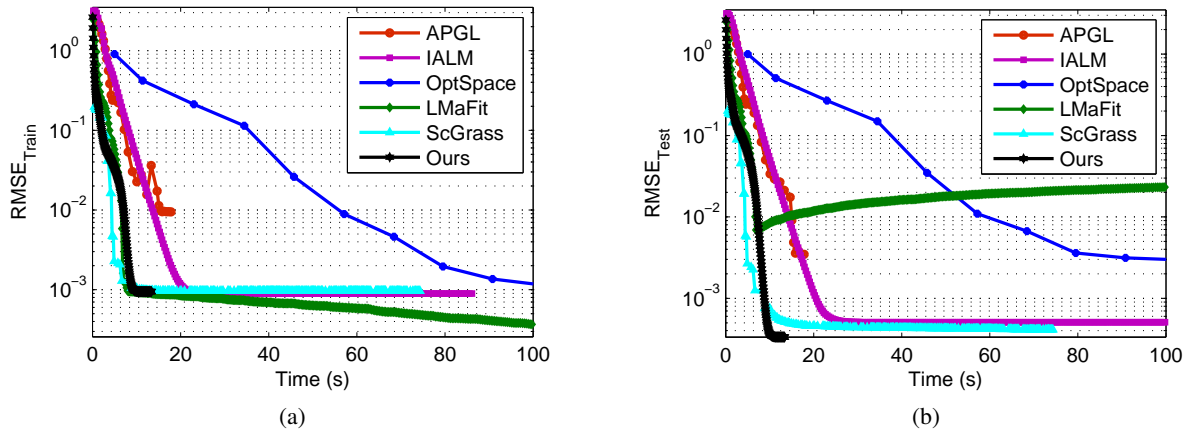


Figure 2: The recovery accuracy on noisy data (where the noise level $nf = 0.001$) vs. running time (seconds): training RMSE (left) and testing RMSE (right).

Figure 2 on noisy matrices of size 2000×2000 , respectively.

We compare our method with APGL¹ [17], IALM² [12], OptSpace³ [10], LMaFit⁴ [22], and ScGrass⁵ [15] on the noiseless or noisy matrices, and illustrate the training and testing recovery accuracies (RMSE) in Figures 1 and 2, respectively, where APGL and IALM use the PROPACK package [11] to compute a partial SVD. All the methods

¹<http://www.math.nus.edu.sg/~mattohkc/NNLS.html>

²<http://www.cis.pku.edu.cn/faculty/vision/zlin/zlin.htm>

³<http://web.engr.illinois.edu/~swoh/software/optspace/>

⁴<http://lmafit.blogs.rice.edu/>

⁵<http://www-users.cs.umn.edu/~thango/>

are implemented in Matlab and use mex files. In terms of running time, the results show that for the noiseless data, our method, LMaFit and ScGrass converge much faster than the other three methods including APGL, IALM, and OptSpace. However, for the noisy data, the testing RMSE of LMaFit becomes worse due to overfitting while the training RMSE gradually decreases.

We also test the robustness of all these methods against the noise, and demonstrate the experimental results (the testing RMSE and running time) in Figure 3. It is clear that when the noise level is higher, our method usually outperforms the other methods in terms of the testing RMSE, that is, our method has the good generation ability. With the increase of the noise level, the running time of the other algorithms dramatically grows except for our method and OptSpace. In contrast, the runtime of our method increases slightly.

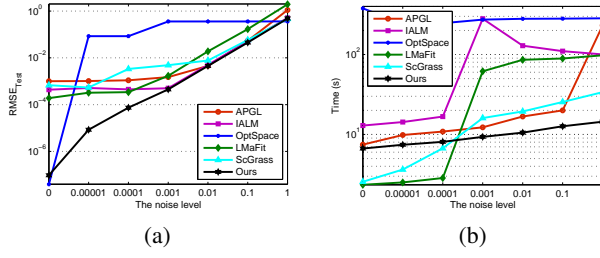


Figure 3: The recovery results vs. the noise level: RMSE (left) and running time (right).

5.2 Real-World Data

In order to evaluate our method, experiments were conducted on three widely used recommendation systems data sets⁶: MovieLens100K (ML-100K) with 100K ratings, MovieLens1M (ML-1M) with 1M ratings, and MovieLens10M (ML-10M) with 10M ratings. We randomly split these three data sets to train and test sets such that the ratio of the train set to test set is 9:1, and the experimental results are reported over 20 independent runs. Except for APGL, IALM, OptSpace, and LMaFit, we also compare our method with two state-of-the-art optimization methods on manifolds: ScGrass and RTRMC⁷ [4]. For our method, we set the rank $d = 5, 6, 7$, and $\mu = 10^{-2}$. The stopping tolerance for all algorithms is set to $\varepsilon = 10^{-4}$. All other parameters are set to their default values for the algorithms that we compare with. We also use the Root Mean Squared Error (RMSE) as the evaluation measure.

The average RMSE on these three data sets is reported over 20 independent runs and is shown in Table 1. The results show that for some fixed ranks, the matrix factorization methods including OptSpace, ScGrass, RTRMC, LMaFit and our method usually perform better than two nuclear norm minimization methods including APGL and IALM. As expected, our method with $d = 5$ on the MovieLens (1M) data set achieved a RMSE of 0.8711, slightly outperforming the well-known restricted Boltzmann machines’s RMSE of 0.8817 [16]. Moreover, our matrix factorization method with nuclear norm regularization consistently outperforms the other matrix factorization methods including OptSpace, ScGrass, RTRMC and LMaFit, and the two nuclear norm minimization methods including APGL and IALM. This confirms that the proposed matrix factorization model with nuclear norm regularization can avoid the over-fitting problems of matrix factorization.

Furthermore, we also analyze the robustness of our method with regard to its parameters: the given rank and the regularization parameter μ on the MovieLens1M data set, as

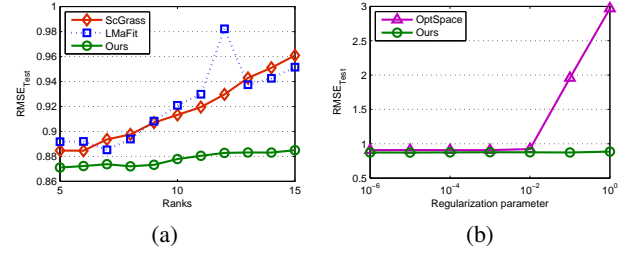


Figure 4: Results of our method with varying parameter values on the MovieLens1M data set.

shown in Figure 4, from which we can see that our method is robust against variations in its parameters. For comparison, we also show the results of two related methods: ScGrass and LMaFit with varying ranks in Figure 4(a). It is clear that, by increasing the number of the given ranks, the RMSE of ScGrass and LMaFit becomes worse. In contrast, the RMSE of our method increases slightly when the number of the given ranks increases. This further confirms that our matrix factorization model with nuclear norm regularization is effective and can avoid overfitting. OptSpace also has a spectral regularization version: $\min_{U, S, V} (1/2) \|\mathcal{P}_\Omega(USV^T - X)\|_F^2 + \mu \|S\|_F^2$. From Figure 4(b), we observe that our method is much more robust than OptSpace in terms of the regularization parameter μ .

Finally, we conduct the running time comparison of all those algorithms on the MovieLens100K and MovieLens1M data sets, as shown in Figure 5. The experiments were performed with Matlab 7.11 on an Intel Core 2 Duo (2.33 GHz) PC running Windows 7 with 2GB main memory. From the results shown in Figure 5, we can observe that our method, ScGrass, RTRMC, and LMaFit are much faster than the other three state-of-the-art algorithms including APGL, IALM and OptSpace. For APGL and IALM, SVD-related calculations essentially dominate their total costs. Therefore, avoiding SVD-related calculations on relative large-scale matrices is a main reason why our method is much faster than the nuclear norm minimization algorithms such as APGL and IALM, validating our original motivation of solving the matrix factorization model with nuclear norm regularization.

5.3 The Impact of Social Context

We also investigate the effects of social context on the MovieLens100K data set, which is suitable to evaluate the impacts of user demographic information and item genre information because it consists of demographic information (e.g. gender, age and occupation) of users and genre of movies. According to [8], a two dimensional feature vector is used to characterize the user’s gender, that is, if the user is male, then the first feature is 1 while the second is 0, and vice versa. The users are partitioned into 7 age group: 1-17,

⁶<http://www.grouplens.org/node/73>

⁷<http://perso.uclouvain.be/nicolas.boumal/RTRMC/>

Table 1: RMSE of different methods on three data sets: MovieLens 100K, MovieLens 1M, and MovieLens 10M.

Methods	MovieLens (100K)			MovieLens (1M)			MovieLens (10M)		
APGL	1.2142			1.1528			0.8637		
IALM	1.2585			1.0153			0.8989		
OptSpace	0.9411			0.9068			1.1357		
Ranks	5	6	7	5	6	7	5	6	7
ScGrass	0.9236	0.9392	0.9411	0.8847	0.8846	0.8936	0.8359	0.8290	0.8247
RTRMC	0.9837	1.0617	1.1642	0.8901	0.8906	0.8977	0.8463	0.8442	0.8386
LMaFit	0.9468	0.9540	0.9568	0.8918	0.8920	0.8853	0.8576	0.8530	0.8423
Ours	0.9216	0.9243	0.9330	0.8711	0.8723	0.8738	0.8330	0.8261	0.8217

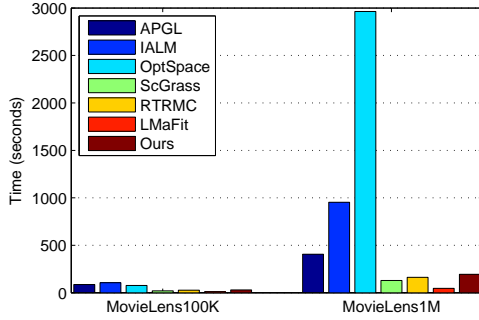


Figure 5: Running time (seconds) for comparison on the MovieLens100K and MovieLens1M data sets.

18-24, 25-34, 35-44, 45-49, 50-55, 56+. Then a seven dimensional feature vector is used to describe the user's age group. In addition, there are totally 21 occupations: administrator, doctor, educator, engineer, entertainment, executive, healthcare, homemaker, lawyer, librarian, marketing, programmer, retired, salesman, scientist, student, technician, writer, other and none. Thus, a 21 dimensional feature vector is used to describe the user's occupation. In total, a 30 dimensional feature vector is achieved for user i . On the other hand, there are 19 genres of movies. Likewise, we use a 19 dimensional feature vector for movie j . We evaluate the impact of user demographic and item genre information on this data set with 60% and 90% training sets, and we report in Fig. 6 the RMSE results yielded by our method without graph regularization, with user or item graph regularization and both graph regularizations. When with the effect of the user demographic or the item genre context, the performance of our method improves. For example, compared with our method without graph regularization, on average, our method with user or item graph regularization have 0.35% and 1.04% relative performance improvement in terms of RMSE, respectively. When with the effects of both the user demographic and the item genre context, our method obtains the best performance, suggesting that the user demographic and the item genre context

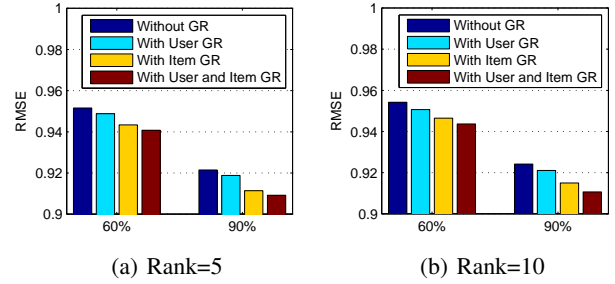


Figure 6: The performance of variants of our method on the the MovieLens100K data set.

contain complementary information to each other for recommendation.

6 Conclusions

In this paper, we proposed a Grassmannian manifold optimization method to tackle the nuclear norm regularized least squares problems with a guarantee of local convergence, such as the noisy matrix completion problem. Our method inherits the superiority of two classes of methods, i.e. soft thresholding approaches and hard thresholding approaches, and has good generation ability. In addition, our method is extended to address the graph regularized problem. We demonstrated with convincing experimental results that our regularized formulation is effective, and our method is robust to noise or against variations in its parameters.

Acknowledgements

We thank the reviewers for their useful comments that have helped improve the paper significantly. This research has been supported by the Hong Kong Research Grants Council (RGC) General Research Fund (GRF) Project Nos. CUHK 411211 and 411310, the Chinese University of Hong Kong Direct Grant Nos. 4055015 and 4055017, and Microsoft Research Asia Grant 6903555.

References

- [1] P.-A. Absil, C.G. Baker, and K. Gallivan. Trust-region methods on riemannian manifolds. *Found. Comput. Math.*, 7(3):303–330, 2007.
- [2] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2008.
- [3] H. Avron, S. Kale, S. Kasiviswanathan, and V. Sindhvani. Efficient and practical stochastic subgradient descent for nuclear norm regularization. In *ICML*, pages 1231–1238, 2012.
- [4] N. Boumal and P.-A. Absil. Rtrmc: A riemannian trust-region method for low-rank matrix completion. In *NIPS*, pages 406–414, 2011.
- [5] J.-F. Cai, E. J. Candes, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SAIM Journal on Optimization*, 20(4):1956–1982, 2010.
- [6] E.J. Candes and T. Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Trans. Inform. Theory*, 56(5):2053–2080, 2010.
- [7] A. Edelman, T.A. Arias, and S.T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM J. Matrix Anal. Appl.*, 20(2):303–353, 1998.
- [8] Q. Gu, Jie Zhou, and C. Ding. Collaborative filtering: weighted nonnegative matrix factorization incorporating user and item graphs. In *SDM*, pages 199–210, 2010.
- [9] S. Ji and J. Ye. An accelerated gradient method for trace norm minimization. In *ICML*, pages 457–464, 2009.
- [10] R.H. Keshavan, S. Oh, and A. Montanari. Matrix completion from a few entries. *IEEE Trans. Inform. Theory*, 56(6):2980–2998, 2010.
- [11] R. Larsen. Propack-software for large and sparse svd calculations. 2004.
- [12] Z. Lin, M. Chen, and L. Wu. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. Technical report, Univ. Illinois, Urbana-Champaign, 2009.
- [13] S. Ma, D. Goldfarb, and L. Chen. Fixed point and bregman iterative methods for matrix rank minimization. *Mathematical Programming*, 128(1):321–353, 2011.
- [14] R. Mazumder, T. Hastie, and R. Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *J. Mach. Learn. Res.*, 11:2287–2322, 2010.
- [15] T. Ngo and Y. Saad. Scaled gradients on grassmann manifolds for matrix completion. In *NIPS*, pages 1421–1429, 2012.
- [16] R. Salakhutdinov, A. Mnih, and G. Hinton. Restricted boltzmann machines for collaborative filtering. In *ICML*, pages 791–798, 2007.
- [17] K.-C. Toh and S. Yun. An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems. *Pacific Journal of Optimization*, 6:615–640, 2010.
- [18] B. Vandereycken. Low-rank matrix completion by riemannian optimization. *SAIM Journal on Optimization*, 23(2):1214–1236, 2013.
- [19] Y. Wang and H. Xu. Stability of matrix factorization for collaborative filtering. In *ICML*, pages 417–424, 2012.
- [20] G.A. Watson. Characterization of the subdifferential of some matrix norms. *Linear Algebra and Its Applications*, 170:33–45, 1992.
- [21] P. Wedin. Perturbation bounds in connection with singular value decomposition. *BIT*, 12:99–111, 1972.
- [22] Z. Wen, W. Yin, and Y. Zhang. Solving a low-rank factorization model for matrix completion by a non-linear successive over-relaxation algorithm. *Mathematical Programming Computation*, 4(4):333–361, 2012.

Fast Ridge Regression with Randomized Principal Component Analysis and Gradient Descent

Yichao Lu ^{*} and Dean P. Foster [†]

Department of Statistics
Wharton, University of Pennsylvania
Philadelphia, PA, 19104-6340

Abstract

We propose a new two stage algorithm LING for large scale regression problems. LING has the same risk as the well known Ridge Regression under the fixed design setting and can be computed much faster. Our experiments have shown that LING performs well in terms of both prediction accuracy and computational efficiency compared with other large scale regression algorithms like Gradient Descent, Stochastic Gradient Descent and Principal Component Regression on both simulated and real datasets.

1 INTRODUCTION

Ridge Regression (RR) is one of the most widely applied penalized regression algorithms in machine learning problems. Suppose \mathbf{X} is the $n \times p$ design matrix and \mathbf{Y} is the $n \times 1$ response vector, ridge regression tries to solve the problem

$$\hat{\beta} = \arg \min_{\beta \in \mathcal{R}^p} \|\mathbf{X}\beta - \mathbf{Y}\|^2 + n\lambda\|\beta\|^2 \quad (1)$$

which has an explicit solution

$$\hat{\beta} = (\mathbf{X}^\top \mathbf{X} + n\lambda)^{-1} \mathbf{X}^\top \mathbf{Y} \quad (2)$$

However, for modern problems with huge design matrix \mathbf{X} , computing (2) costs $O(np^2)$ FLOPS. When $p > n \gg 1$ one can consider the dual formulation of (1) which also has an explicit solution as mentioned in (Lu et al., 2013; Saunders et al., 1998) and the cost is $O(n^2p)$ FLOPS. In summary, trying to solve (1) exactly costs $O(np \min\{n, p\})$ FLOPS which can be very slow.

There are faster ways to approximate (2) when computational cost is the concern. One can view RR as an optimization problem and use Gradient Descent (GD) which

takes $O(np)$ FLOPS in every iteration. However, the convergence speed for GD depends on the spectrum of \mathbf{X} and the magnitude of λ . When \mathbf{X} is ill conditioned and λ is small, GD requires a huge number of iterations to converge which makes it very slow. For huge datasets, one can also apply stochastic gradient descent (SGD) (Zhang, 2004; Johnson and Zhang, 2013; Bottou, 2010), a powerful tool for solving large scale optimization problems.

Another alternative for regression on huge datasets is Principal Component Regression (PCR) as mentioned in (Artemiou and Li, 2009; Jolliffe, 2005), which runs regression only on the top k_1 principal components (PCs) of the \mathbf{X} matrix. PCA for huge \mathbf{X} can be computed efficiently by randomized algorithms like (Halko et al., 2011a,b). The cost for computing top k_1 PCs of \mathbf{X} is $O(npk_1)$ FLOPS. The connection between RR and PCR is well studied by (Dhillon et al., 2013). The problem of PCR is that when a large proportion of signal sits on the bottom PCs, it has to regress on a lot of PCs which makes it both slow and inaccurate (see later sections for detailed explanations).

In this paper, we propose a two stage algorithm LING¹ which is a faster way of computing the RR solution (2). A detailed description of the algorithm is given in section 2. In section 3, we prove that LING has the same risk as RR under the fixed design setting. In section 4, we compare the performance of PCR, GD, SGD and LING in terms of prediction accuracy and computational efficiency on both simulated and real data sets.

2 THE ALGORITHM

2.1 DESCRIPTION OF THE ALGORITHM

LING is a two stage algorithm. The intuition of LING is quite straight forward. We start with the observation that regressing \mathbf{Y} on \mathbf{X} (OLS) is essentially projecting \mathbf{Y} onto the span of \mathbf{X} . Let \mathbf{U}_1 denote the top k_2 PCs (left singular vectors) of \mathbf{X} and let \mathbf{U}_2 denote the remaining PCs. The projection of \mathbf{Y} onto the span of \mathbf{X} can be decom-

^{*} yichaolu@wharton.upenn.edu

[†] foster@wharton.upenn.edu

¹LING is the Chinese of ridge

Algorithm 1 LING

Input : Data matrix \mathbf{X}, \mathbf{Y} . \mathbf{U}_1 , an orthonormal matrix consists of top k_2 PCs of \mathbf{X} . d_1, d_2, \dots, d_{k_2} , top k_2 singular values of \mathbf{X} . Regularization parameter λ , an initial vector $\hat{\gamma}_{2,0}$ and number of iterations n_2 for GD.

Output : $\hat{\gamma}_{1,s}, \hat{\gamma}_2$, the regression coefficients.

1. Regress \mathbf{Y} on \mathbf{U}_1 , let $\hat{\gamma}_1 = \mathbf{U}_1^\top \mathbf{Y}$.
 2. Compute the residual of previous regression problem. Let $\mathbf{Y}_r = \mathbf{Y} - \mathbf{U}_1 \hat{\gamma}_1$.
 3. Compute the residual of \mathbf{X} regressing on \mathbf{U}_1 . Use $\mathbf{X}_r = \mathbf{X} - \mathbf{U}_1 \mathbf{U}_1^\top \mathbf{X}$ to denote the residual of \mathbf{X} .
 4. Use gradient descent with optimal step size with initial value $\hat{\gamma}_{2,0}$ (see algorithm 3) to solve the RR problem $\min_{\hat{\gamma}_2 \in \mathcal{R}^p} \|\mathbf{X}_r \hat{\gamma}_2 - \mathbf{Y}_r\|^2 + n\lambda \|\hat{\gamma}_2\|^2$.
 5. Compute a shrinkage version of $\hat{\gamma}_1$ by $(\hat{\gamma}_{1,s})_i = \frac{d_i^2}{d_i^2 + n\lambda} (\hat{\gamma}_1)_i$
 6. The final estimator is $\hat{\mathbf{Y}} = \mathbf{U}_1 \hat{\gamma}_{1,s} + \mathbf{X}_r \hat{\gamma}_2$.
-

posed into two orthogonal parts, the projection onto \mathbf{U}_1 and the projection onto \mathbf{U}_2 . In the first stage, we pick a $k_2 \ll p$ and the projection onto \mathbf{U}_1 can be computed directly by $\hat{\mathbf{Y}}_1 = \mathbf{U}_1 \mathbf{U}_1^\top \mathbf{Y}$ which is exactly the same as running a PCR on top k_2 PCs. For huge \mathbf{X} , computing the top k_2 PCs exactly is very slow, so we use a faster randomized SVD algorithm for computing \mathbf{U}_1 which is proposed by (Halko et al., 2011a) and described below. In the second stage, we first compute $\mathbf{Y}_r = \mathbf{Y} - \hat{\mathbf{Y}}_1$ and $\mathbf{X}_r = \mathbf{X} - \mathbf{U}_1 \mathbf{U}_1^\top \mathbf{X}$ which are the residual of \mathbf{Y} and \mathbf{X} after projecting onto \mathbf{U}_1 . Then we compute the projection of \mathbf{Y} onto the span of \mathbf{U}_2 by solving the optimization problem $\min_{\hat{\gamma}_2 \in \mathcal{R}^p} \|\mathbf{X}_r \hat{\gamma}_2 - \mathbf{Y}_r\|^2$ with GD (Algorithm 3). Finally, since RR shrinks the projection of \mathbf{Y} onto \mathbf{X} (the OLS solution) via regularization, we also shrink the projections in both stages accordingly. Shrinkage in the first stage is performed directly on the estimated regression coefficients and shrinkage in the second stage is performed by adding a regularization term to the optimization problem mentioned above. Detailed description of LING is shown in Algorithm 1.

Remark 1. LING can be regarded as a combination of PCR and GD. The first stage of LING is a very crude estimation of \mathbf{Y} and the second stage adds a correction to the first stage estimator. Since we do not need a very accurate estimator in the first stage it suffices to pick a very small k_2 in contrast with the k_1 PCs needed for PCR. In the second stage, the design matrix \mathbf{X}_r is a much better conditioned matrix than the original \mathbf{X} since the directions with largest singular values are removed. As introduced in section 2.2, Algorithm 3 converges much faster with a better conditioned matrix. Hence GD in the second stage of LING converges faster than directly applying GD for solving (1). The above property guarantees that LING is both fast and accurate compared with PCR and GD. More details about

Algorithm 2 Random SVD

Input : design matrix \mathbf{X} , target dimension k_2 , number of power iterations i .

Output : $\mathbf{U}_1 \in n \times k_2$, the matrix of top k_2 left singular vectors of \mathbf{X} , d_1, d_2, \dots, d_{k_2} , the top k_2 singular values of \mathbf{X} .

1. Generate random matrix $R_1 \in p \times k_2$ with i.i.d standard Gaussian entries.
 2. Estimate the span of top k_2 left singular vectors of \mathbf{X} by $A_1 = (\mathbf{X}\mathbf{X}^\top)^i \mathbf{X} R_1$.
 3. Use QR decomposition to compute Q_1 which is an orthonormal basis of the column space of A_1 .
 4. Compute SVD of the reduced matrix $Q_1^\top \mathbf{X} = \mathbf{U}_0 \mathbf{D}_0 \mathbf{V}_0^\top$.
 5. $\mathbf{U}_1 = Q_1 \mathbf{U}_0$ gives the top k_2 singular vectors of \mathbf{X} and the diagonal elements of \mathbf{D}_0 gives the singular values.
-

Algorithm 3 Gradient Descent with Optimal Step Size (GD)

Goal : Solve the ridge problem $\min_{\hat{\gamma} \in \mathcal{R}^p} \|\mathbf{X}\hat{\gamma} - \mathbf{Y}\|^2 + n\lambda \|\hat{\gamma}\|^2$.

Input : Data matrix \mathbf{X}, \mathbf{Y} , regularization parameter λ , number of iterations n_2 , an initial vector $\hat{\gamma}_0$

Output : $\hat{\gamma}$

for $t = 0$ **to** $n_2 - 1$ **do**

$$Q = 2\mathbf{X}^\top \mathbf{X} + 2n\lambda I$$

$$w_t = 2\mathbf{X}^\top \mathbf{Y} - Q\hat{\gamma}_t$$

$$s_t = \frac{w_t^\top w_t}{w_t^\top Q w_t}. \quad s_t \text{ is the step size which makes the target function decrease the most in direction } w_t.$$

$$\hat{\gamma}_{t+1} = \hat{\gamma}_t + s_t \cdot w_t.$$

end for

on the computational cost will be discussed in section 2.2.

Remark 2. Algorithm 2 is essentially approximating the subspace of top left singular vectors by random projection. It provides a fast approximation of the top singular values and vectors for large \mathbf{X} when computing the exact SVD is very slow. Theoretical guarantees and more detailed explanations can be found in (Halko et al., 2011a). Empirically we find in the experiments, Algorithm 2 may occasionally generate a bad subspace estimator due to randomness which makes PCR perform badly. On the other hand, LING is much more robust since in the second stage it compensate for the signal missing in the first stage. In all the experiments, we set $i = 1$.

The shrinkage step (step 5) in Algorithm 1 is only necessary for theoretical purposes since the goal is to approximate Ridge Regression which shrinks the Least Square estimator over all directions. In practice shrinkage over the top k_2 PCs is not necessary. Usually the number of PCs selected (k_2) is very small. From the bias variance trade off perspective, the variance reduction gained from the shrinkage over top k_2 PCs is at most $O(\frac{k_2}{n})$ under

the fixed design setting (Dhillon et al., 2013) which is a tiny number. Moreover, since the top singular values of $\mathbf{X}^\top \mathbf{X}$ are usually very large compared with $n\lambda$ in most real problems, the shrinkage factor $\frac{d_i^2}{d_i^2 + n\lambda}$ will be pretty close to 1 for top singular values. We use shrinkage in Algorithm 1 because the risk of the shrinkage version of LING is exactly the same as RR as proved in section 3.

Algorithm 2 can be further simplified if we skip the shrinkage step mentioned in previous paragraph. Instead of computing the top k_2 PCs, the only thing we need to know is the subspace spanned by these PCs since the first stage is essentially projecting \mathbf{Y} onto this subspace. In other words, we can replace \mathbf{U}_1 in step 1, 2, 3 of Algorithm 1 with Q_1 obtained in step 3 of Algorithm 2 and directly let $\hat{\mathbf{Y}} = Q_1 \hat{\gamma}_1 + \mathbf{X}_r \hat{\gamma}_2$. In the experiments of section 4 we use this simplified version.

2.2 COMPUTATIONAL COST

We claim that the cost of LING is $O(np(k_2 + n_2))$ where k_2 is the number of PCs used in the first stage and n_2 is the number of iterations of GD in the second stage. According to (Halko et al., 2011a), the dominating step in Algorithm 2 is computing $(\mathbf{X}\mathbf{X}^\top)^k \mathbf{X} R_1$ and $Q_1^\top \mathbf{X}$ which costs $O(npk_2)$ FLOPS. Computing $\hat{\gamma}_1$ and \mathbf{Y}_r cost less than $O(npk_2)$. Computing \mathbf{X}_r costs $O(npk_2)$. So the computational cost before the GD step is $O(npk_2)$. For the GD stage, note that in every iteration Q never needs to be constructed explicitly. While computing w_t and s_t , always multiplying matrix and vector first gives a cost of $O(np)$ for every iteration. So the cost for GD stage is $O(n_2 np)$. Add all pieces together the cost of LING is $O(np(k_2 + n_2))$ FLOPS.

Let n_1 be the number of iterations required for solving (1) directly by GD and k_1 be the number of PCs used for PCR. It's easy to check that the cost for GD is $O(n_1 np)$ FLOPS and the cost for PCR is $O(npk_1)$. As mentioned in remark 1, the advantage of LING over GD and PCR is that k_1 and n_1 might have to be really large to achieve high accuracy but much smaller values of the pair (k_2, n_2) will work for LING.

In the remaining part of the paper we use "signal on certain PCs" to refer to the projection of \mathbf{Y} onto certain principal components of \mathbf{X} . Consider the case when the signal is widely spread among all PCs (i.e. the projection of \mathbf{Y} onto the bottom PCs of \mathbf{X} is not very small) instead of concentrating on the top ones, k_1 needs to be large to make PCR perform well since the signal on bottom PCs are discarded by PCR. But LING does not need to include all the signal in the first stage regression since the signal left over will be estimated in the second GD stage. Therefore LING is able to recover most of the signal even with a small k_2 .

In order to understand the connection between accuracy and number of iterations in Algorithm 3, we state the fol-

lowing theorem in A.Epelman (2007):

Theorem 1. Let $g(z) = \frac{1}{2} z^\top M z + q^\top z$ be a quadratic function where M is a PSD matrix. Suppose $g(z)$ achieves minimum at z^* . Apply Algorithm 3 to solve the minimization problem. Let z_t be the z value after t iterations, then the gap between $g(z_t)$ and $g(z^*)$, the minimum of the objective function satisfies

$$\frac{g(z_{t+1}) - g(z^*)}{g(z_t) - g(z^*)} \leq C = \left(\frac{A - a}{A + a} \right)^2 \quad (3)$$

where A, a are the largest and smallest eigenvalue of the M matrix.

Theorem 1 shows that the sub optimality of the target function decays exponentially as the number of iterations increases and the speed of decay depends on the largest and smallest singular value of the PSD matrix that defines the quadratic objective function. If we directly apply GD to solve (1), Let $f_1(\beta) = \|\mathbf{X}\beta - \mathbf{Y}\|^2 + n\lambda\|\beta\|^2$. Assume f_1 reaches its minimum at $\hat{\beta}$. Let $\hat{\beta}_t$ be the coefficient after t iterations and let d_i denote the i^{th} singular value of \mathbf{X} . Applying theorem 1, we have

$$\frac{f_1(\hat{\beta}_{t+1}) - f_1(\hat{\beta})}{f_1(\hat{\beta}_t) - f_1(\hat{\beta})} \leq C = \left(\frac{d_1^2 - d_p^2}{d_1^2 + d_p^2 + 2n\lambda} \right)^2 \quad (4)$$

Similarly for the second stage of LING, Let $f_2(\gamma_2) = \|\mathbf{X}_r \gamma_2 - \mathbf{Y}_r\|^2 + n\lambda\|\gamma_2\|^2$. Assume f_2 reaches its minimal at $\hat{\gamma}_2$. We have

$$\frac{f_2(\hat{\gamma}_{2,t+1}) - f_2(\hat{\gamma}_2)}{f_2(\hat{\gamma}_{2,t}) - f_2(\hat{\gamma}_2)} \leq C = \left(\frac{d_{k_2+1}^2}{d_{k_2+1}^2 + 2n\lambda} \right)^2 \quad (5)$$

In most real problems, the top few singular values of $\mathbf{X}^\top \mathbf{X}$ are much larger than the other singular values and $n\lambda$. Therefore the constant C obtained in (4) can be very close to 1 which makes GD algorithm converges very slowly. On the other hand, removing the top few PCs will make C in (5) significantly smaller than 1. In other words, GD may take a lot of iterations to converge when solving (1) directly while the second stage of LING takes much less iterations to converge. This can also be seen in the experiments of section 4.

3 THEOREMS

In this section we compute the risk of LING estimator (explained below) under the fixed design setting. For simplicity, assume $\mathbf{U}_1, \mathbf{D}_0$ generated by Algorithm 2 give exactly the top k_2 left singular vectors and singular values of \mathbf{X} and GD in step 4 of Algorithm 1 converges to the optimal solution. Let $\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$ be the SVD of \mathbf{X} where $\mathbf{U} = (\mathbf{U}_1, \mathbf{U}_2)$ and $\mathbf{V} = (\mathbf{V}_1, \mathbf{V}_2)$. Here $\mathbf{U}_1, \mathbf{V}_1$ are top k_2 singular vectors and $\mathbf{U}_2, \mathbf{V}_2$ are bottom

$p - k_2$ singular vectors. Let $\mathbf{D} = \text{diag}(\mathbf{D}_1, \mathbf{D}_2)$ where $\mathbf{D}_1 \in k_2 \times k_2$ contains top k_2 singular values denoted by $d_1 \geq d_2 \geq \dots \geq d_{k_2}$ and $\mathbf{D}_2 \in p - k_2 \times p - k_2$ contains bottom $p - k_2$ singular values. Let $\mathbf{D}_3 = \text{diag}(\mathbf{0}, \mathbf{D}_2)$ (replace \mathbf{D}_1 in \mathbf{D} by a zero matrix of the same size).

3.1 THE FIXED DESIGN MODEL

Assume \mathbf{X} , \mathbf{Y} comes from the fixed design model $\mathbf{Y} = \mathbf{X}\beta + \epsilon$ where $\epsilon \in n \times 1$ are i.i.d noise with mean 0 and variance σ^2 . Here \mathbf{X} is fixed and the randomness of \mathbf{Y} only comes from ϵ . Note that $\mathbf{X} = \mathbf{U}_1 \mathbf{D}_1 \mathbf{V}_1^\top + \mathbf{X}_r$, the fixed design model can also be written as

$$\mathbf{Y} = (\mathbf{U}_1 \mathbf{D}_1 \mathbf{V}_1^\top + \mathbf{X}_r) \beta + \epsilon = \mathbf{U}_1 \gamma_1 + \mathbf{X}_r \gamma_2 + \epsilon$$

where $\gamma_1 = \mathbf{D}_1 \mathbf{V}_1^\top \beta$ and $\gamma_2 = \beta$. We use the l_2 distance between $\mathbb{E}(\mathbf{Y}|\mathbf{X})$ (the best possible prediction given \mathbf{X} under l_2 loss) and $\hat{\mathbf{Y}} = \mathbf{U}_1 \hat{\gamma}_{1,s} + \mathbf{X}_r \hat{\gamma}_2$ (the prediction by LING) as the loss function, which is called risk in the following discussions. Actually $\mathbb{E}(\mathbf{Y}|\mathbf{X}) = \mathbf{X}\beta$ is linear in \mathbf{X} under fixed design model. The risk of LING can be written as

$$\begin{aligned} & \frac{1}{n} \mathbb{E} \|\mathbb{E}(\mathbf{Y}|\mathbf{X}) - \mathbf{U}_1 \hat{\gamma}_{1,s} - \mathbf{X}_r \hat{\gamma}_2\|^2 \\ &= \frac{1}{n} \mathbb{E} \|\mathbf{U}_1 \gamma_1 + \mathbf{X}_r \gamma_2 - \mathbf{U}_1 \hat{\gamma}_{1,s} - \mathbf{X}_r \hat{\gamma}_2\|^2 \end{aligned}$$

We can further decompose the risk into two terms:

$$\begin{aligned} & \frac{1}{n} \mathbb{E} \|\mathbf{U}_1 \gamma_1 + \mathbf{X}_r \gamma_2 - \mathbf{U}_1 \hat{\gamma}_{1,s} - \mathbf{X}_r \hat{\gamma}_2\|^2 = \\ & \frac{1}{n} \mathbb{E} \|\mathbf{U}_1 \gamma_1 - \mathbf{U}_1 \hat{\gamma}_{1,s}\|^2 + \frac{1}{n} \mathbb{E} \|\mathbf{X}_r \gamma_2 - \mathbf{X}_r \hat{\gamma}_2\|^2 \end{aligned} \quad (6)$$

because $\mathbf{U}_1^\top \mathbf{X}_r = 0$. Note that here the expectation is taken with respect to ϵ .

Let's calculate the two terms in (6) separately. For the first term we have:

Lemma 1.

$$\frac{1}{n} \mathbb{E} \|\mathbf{U}_1 \gamma_1 - \mathbf{U}_1 \hat{\gamma}_{1,s}\|^2 = \frac{1}{n} \sum_{j=1}^{k_2} \frac{d_j^4 \sigma^2 + \gamma_{1,j}^2 n^2 \lambda^2}{(d_j^2 + n\lambda)^2} \quad (7)$$

Here $\gamma_{1,j}$ is the j^{th} element of γ_1 .

Proof. Let $S \in k_2 \times k_2$ be the diagonal matrix with $S_{j,j} = \frac{d_j^2}{d_j^2 + n\lambda}$. So we have $\hat{\gamma}_{1,s} = S \mathbf{U}_1^\top \mathbf{Y} = S \gamma_1 + S \mathbf{U}_1^\top \epsilon$,

$$\mathbb{E}(\hat{\gamma}_{1,s}) = S \gamma_1.$$

$$\begin{aligned} & \frac{1}{n} \mathbb{E} \|\mathbf{U}_1 \gamma_1 - \mathbf{U}_1 \hat{\gamma}_{1,s}\|^2 \\ &= \frac{1}{n} \mathbb{E} \|\mathbf{U}_1 \mathbb{E}(\hat{\gamma}_{1,s}) - \mathbf{U}_1 \hat{\gamma}_{1,s}\|^2 \\ & \quad + \frac{1}{n} \|\mathbf{U}_1 \gamma_1 - \mathbf{U}_1 \mathbb{E}(\hat{\gamma}_{1,s})\|^2 \\ &= \frac{1}{n} \mathbb{E} \|\mathbf{U}_1 S \mathbf{U}_1^\top \epsilon\|^2 + \frac{1}{n} \|\gamma_1 - S \gamma_1\|^2 \\ &= \frac{1}{n} \mathbb{E} \text{Tr}(\mathbf{U}_1 S^2 \mathbf{U}_1^\top \epsilon \epsilon^\top) + \frac{1}{n} \|\gamma_1 - S \gamma_1\|^2 \\ &= \frac{1}{n} \mathbb{E} \text{Tr}(S^2) \sigma^2 + \frac{1}{n} \|\gamma_1 - S \gamma_1\|^2 \\ &= \frac{1}{n} \sum_{j=1}^{k_2} \frac{d_j^4 \sigma^2 + \gamma_{1,j}^2 n^2 \lambda^2}{(d_j^2 + n\lambda)^2} \end{aligned}$$

□

Now consider the second term in (6).

Note that

$$\mathbf{X}_r = \mathbf{U} \mathbf{D}_3 \mathbf{V}^\top$$

The residual \mathbf{Y}_r after the first stage can be represented by

$$\mathbf{Y}_r = \mathbf{Y} - \mathbf{U}_1 \hat{\gamma}_1 = (I - \mathbf{U}_1 \mathbf{U}_1^\top) \mathbf{Y} = \mathbf{X}_r \gamma_2 + (I - \mathbf{U}_1 \mathbf{U}_1^\top) \epsilon$$

and the optimal coefficient obtained in the second GD stage is

$$\hat{\gamma}_2 = (\mathbf{X}_r^\top \mathbf{X}_r + n\lambda I)^{-1} \mathbf{X}_r^\top \mathbf{Y}_r$$

For simplicity, let $\epsilon_2 = (I - \mathbf{U}_1 \mathbf{U}_1^\top) \epsilon$.

Lemma 2.

$$\mathbb{E} \|\mathbf{X}_r \gamma_2 - \mathbf{X}_r \hat{\gamma}_2\|^2 = \sum_{i=k_2+1}^p \frac{1}{(d_i^2 + n\lambda)^2} (d_i^4 \sigma^2 + n\lambda^2 d_i^2 \alpha_i^2) \quad (8)$$

where α_i is the i^{th} element of $\alpha = \mathbf{V}^\top \gamma_2$

Proof. First define

$$\begin{aligned} \mathbf{X}_\lambda &= \mathbf{X}_r^\top \mathbf{X}_r + n\lambda I \\ \mathbf{D}_\lambda &= \mathbf{D}_3^2 + n\lambda I \end{aligned}$$

$$\begin{aligned} \mathbb{E} \|\mathbf{X}_r \gamma_2 - \mathbf{X}_r \hat{\gamma}_2\|^2 &= \|\mathbf{X}_r \gamma_2 - \mathbf{X}_r \mathbb{E}(\hat{\gamma}_2)\|^2 \quad (9) \\ & \quad + \mathbb{E} \|\mathbf{X}_r \mathbb{E}(\hat{\gamma}_2) - \mathbf{X}_r \hat{\gamma}_2\|^2 \quad (10) \end{aligned}$$

Consider (9) and (10) separately.

$$\begin{aligned} (9) &= \|\mathbf{X}_r \mathbf{X}_\lambda^{-1} \mathbf{X}_r^\top \mathbf{X}_r \gamma_2 - \mathbf{X}_r \gamma_2\|^2 \\ &= \|\mathbf{U} \mathbf{D}_3 \mathbf{D}_\lambda^{-1} \mathbf{D}_3^\top \mathbf{V}^\top \gamma_2 - \mathbf{U} \mathbf{D}_3 \mathbf{V}^\top \gamma_2\|^2 \\ &= \|\mathbf{D}_3 \mathbf{D}_\lambda^{-1} \mathbf{D}_3^\top \alpha - \mathbf{D}_3 \alpha\|^2 \\ &= \sum_{i=k_2+1}^p \alpha_i^2 d_i^2 \left(\frac{n\lambda}{d_i^2 + n\lambda} \right)^2 \end{aligned}$$

$$\begin{aligned}
(10) &= \mathbb{E}_{\epsilon_2} \|\mathbf{X}_r \mathbf{X}_\lambda^{-1} \mathbf{X}_r^\top \epsilon_2\|^2 \\
&= \mathbb{E}_{\epsilon_2} \text{Tr}(\mathbf{X}_r \mathbf{X}_\lambda^{-1} \mathbf{X}_r^\top \mathbf{X}_r \mathbf{X}_\lambda^{-1} \mathbf{X}_r^\top \epsilon_2 \epsilon_2^\top) \\
&= \mathbb{E}_{\epsilon_2} \text{Tr}(\mathbf{D}_3 \mathbf{D}_\lambda^{-1} \mathbf{D}_3^2 \mathbf{D}_\lambda^{-1} \mathbf{D}_3 \mathbf{U}^\top \epsilon_2 \epsilon_2^\top \mathbf{U}) \\
&= \text{Tr}(\mathbf{D}_3 \mathbf{D}_\lambda^{-1} \mathbf{D}_3^2 \mathbf{D}_\lambda^{-1} \mathbf{D}_3 \mathbb{E}_{\epsilon_2} [\mathbf{U}^\top \epsilon_2 \epsilon_2^\top \mathbf{U}])
\end{aligned}$$

Note that

$$\mathbb{E}_{\epsilon_2} [\mathbf{U}^\top \epsilon_2 \epsilon_2^\top \mathbf{U}] = \text{diag}(0, I_{p-k_2}) \sigma^2$$

($\text{diag}(0, I_{p-k_2})$) replace the top $k_2 \times k_2$ block of the identity matrix with 0),

$$(10) = \sum_{i=k_2+1}^p \frac{d_i^4}{(d_i^2 + n\lambda)^2} \sigma^2 \quad (11)$$

Add the two terms together finishes the proof. \square

Plug (7) (8) into (6) we have

Theorem 2. *The risk of LING algorithm under fixed design setting is*

$$\frac{1}{n} \sum_{j=1}^{k_2} \frac{d_j^4 \sigma^2 + \gamma_{1,j}^2 n^2 \lambda^2}{(d_j^2 + n\lambda)^2} + \frac{1}{n} \sum_{i=k_2+1}^p \frac{d_i^4 \sigma^2 + n^2 \lambda^2 d_i^2 \alpha_i^2}{(d_i^2 + n\lambda)^2} \quad (12)$$

Remark 3. *This risk is the same as the risk of ridge regression provided by Lemma 1 in (Dhillon et al., 2013). Actually, LING gets exactly the same prediction as RR on the training dataset. This is very intuitive since on the training set LING is essentially decomposing the RR solution into the first stage shrinkage PCR predictor on top k_2 PCs and the second stage GD predictor over the residual spaces as explained in section 2.*

4 EXPERIMENTS

In this section we compare the accuracy and computational cost (evaluated in terms of FLOPS) of 3 different algorithms for solving Ridge Regression: Gradient Descent with Optimal step size (GD), Stochastic Variance Reduction Gradient (SVRG) (Johnson and Zhang, 2013) and LING. Here SVRG is an improved version of stochastic gradient descent which achieves exponential convergence with constant step size. We also consider Principal Component Regression (PCR) (Artemiou and Li, 2009; Jolliffe, 2005) which is another common way for running large scale regression. Experiments are performed on 3 simulated models and 2 real datasets. In general, LING performs well on all 3 simulated datasets while GD, SVRG and PCR fails in some cases. For two real datasets, all algorithms give reasonable performance while SVRG and LING are the best. Moreover, both stages of LING require only a moderate amount of matrix multiplications each cost $O(np)$, much faster to run on matlab compared with SVRG which contains a lot of loops.

Table 1: parameter setup for simulated data

	MODEL 1	MODEL 2	MODEL 3
k_1	21,22,23,26 30,50,100	20,30,50 100,150,400	20,30,50,100 150,400
n_1	10,20,30 50,80,100 150,200	2,4,6,8,10 15,20,30	6,10,15,20 30,50,80 120,180,250
k_2	20	20	20
n_2	1,2,3,5 8,13,20	2,4,6,8,10 15,20,30	2,4,6,8,10 15,30
n_{SVRG}	30,50,80 120,150	5,10,20 30,50	5,10,15,25 40,60,90

4.1 SIMULATED DATA

Three different datasets are constructed based on the fixed design model $\mathbf{Y} = \mathbf{X}\beta + \epsilon$ where \mathbf{X} is of size 2000×1500 . In the three experiments \mathbf{X} and β are generated randomly in different ways (more details in following sections) and i.i.d Gaussian noise is added to $\mathbf{X}\beta$ to get \mathbf{Y} . Then GD, SVRG, PCR and LING are performed on the dataset. For GD, we try different number of iterations n_1 . For SVRG, we vary the number of passes through data denoted by n_{SVRG} . The number of iterations SVRG takes equals the number of passes through data times sample size and each iteration takes $O(p)$ FLOPS. The step size of SVRG is chosen by cross validation but this cost is not considered when evaluating the total computational cost. Note that one advantage of GD and LING is that due to the simple quadratic form of the target function, their step size can be computed directly from the data without cross validation which introduces extra cost. For PCR we pick different number of PCs (k_1). For LING we pick top k_2 PCs in the first stage and try different number of iterations n_2 in the second stage. The computational cost and the risk of the four algorithms are computed. The above procedure is repeated over 20 random generation of \mathbf{X} , β and \mathbf{Y} . The risk and computational cost of the traditional RR solution (2) for every dataset is also computed as a benchmark.

The parameter set up for the three datasets are listed in table 1.

4.1.1 MODEL 1

In this model the design matrix \mathbf{X} has a steep spectrum. The top 30 singular values of \mathbf{X} decay exponentially as 1.3^i where $i = 40, 39, 38, \dots, 11$. The spectrum of \mathbf{X} is shown in figure 4. To generate \mathbf{X} , we fix the diagonal matrix \mathbf{D}_e with the designed spectrum and construct \mathbf{X} by $\mathbf{X} = \mathbf{U}_e \mathbf{D}_e \mathbf{V}_e^\top$ where $\mathbf{U}_e, \mathbf{V}_e$ are two random orthonormal matrices. The elements of β are sampled uniformly from interval $[-2.5, 2.5]$. Under this set up, most of the energy of the \mathbf{X} matrix lies in top PCs since the top singular values are much larger than the remaining ones so PCR works well. But as indicated by (4), the convergence of GD

is very slow.

The computational cost and average risk of the four algorithms and also the RR solution (2) over 20 repeats are shown in figure 1. As shown in figure 1 both PCR and LING work well by achieving risk close to RR at less computational cost. SVRG is worse than PCR and LING but much better than GD.

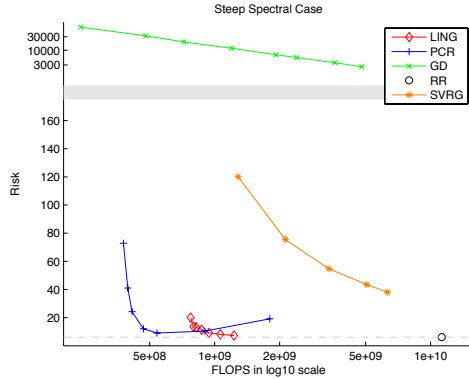


Figure 1: Model 1, Risk VS. Computational Cost plot. PCR and LING approaches the RR risk very fast. SVRG also approaches RR risk but cost more than the previous two. GD is very slow and inaccurate.

4.1.2 MODEL 2

In this model the design matrix \mathbf{X} has a flat spectrum. The singular values of \mathbf{X} are sampled uniformly from $[\frac{\sqrt{2000}}{2}, \sqrt{2000}]$. The spectrum of \mathbf{X} is shown in figure 5. To generate \mathbf{X} , we fix the diagonal matrix \mathbf{D}_e with the designed spectrum and construct \mathbf{X} by $\mathbf{X} = \mathbf{U}_e \mathbf{D}_e \mathbf{V}_e^T$ where $\mathbf{U}_e, \mathbf{V}_e$ are two random orthonormal matrices. The elements of β are sampled uniformly from interval $[-2.5, 2.5]$. Under this set up, the signal are widely spread among all PCs since the spectrum of \mathbf{X} is relatively flat. PCR breaks down because it fails to catch the signal on bottom PCs. As indicated by (4), GD converges relatively fast due to the flat spectrum of \mathbf{X} .

The computational cost and average risk of the four algorithms and also the RR solution (2) over 20 repeats are shown in figure 2. As shown by the figure GD works best since it approaches the risk of RR at the the lowest computational cost. LING and SVRG also works by achieving reasonably low risk with less computational cost. PCR works poorly as explained before.

4.1.3 MODEL 3

This model presented a special case where both PCR and GD will break down. The singular values of $\mathbf{X} \in 2000 \times 1500$ are constructed by first uniformly sample from $[\frac{\sqrt{2000}}{2}, \sqrt{2000}]$. The top 15 sampled values are then multiplied by 10. The top 100 singular values of \mathbf{X} are shown in

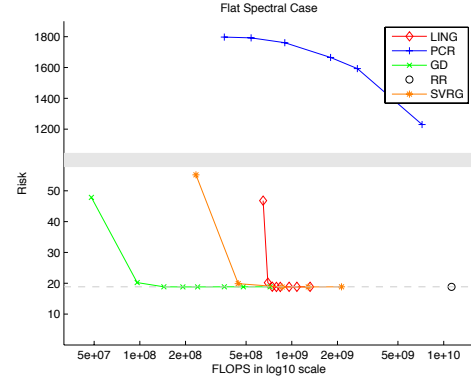


Figure 2: Model 2, Risk VS. Computational Cost plot. GD approaches the RR risk very fast. SVRG and LING are slower than GD but still achieves risk close to RR at less cost. PCR is slow and has huge risk.

figure 6. To generate \mathbf{X} , we fix the diagonal matrix \mathbf{D}_e with the designed spectrum and construct \mathbf{X} by $\mathbf{X} = \mathbf{U}_e \mathbf{D}_e$ where \mathbf{U}_e is a random orthonormal matrix. The first 15 and last 1000 elements of the coefficient vector $\beta \in 1500 \times 1$ are sampled uniformly from interval $[-2.5, 2.5]$ and other elements of β remains 0. In this set up, \mathbf{X} has orthogonal columns which are the PCs, and the signal lies only on the top 15 and bottom 1000 PCs. PCR won't work since a large proportion of signal lies on the bottom PCs. On the other hand, GD won't work as well since the top few singular values are too large compared with other singular values, which makes GD converges very slowly.

The computational cost and risk of the four algorithms and also the RR solution (2) over 20 repeats are shown in figure 3. As shown by the figure LING works best in this set up. SVRG is slightly worse than LING but still approaching RR with less cost. In this case, GD converges slowly and PCR is completely off target as explained before.

4.2 REAL DATA

In this section we compare PCR, GD, SVRG and LING with the RR solution (2) on two real datasets.

4.2.1 GISETTE DATASET

The first is the gisette data set (Guyon et al., 2004) from the UCI repository which is a bi-class classification task. Every row of the design matrix $\mathbf{X} \in 6000 \times 5000$ consists of pixel features of a single digit "4" or "9" and \mathbf{Y} gives the class label. Among the 6000 samples, we use 5000 for training and 1000 for testing. The classification error rate for RR solution (2) is 0.019. Since the goal is to compare different algorithms for regression, we don't care about achieving the state of the art accuracy for this dataset as long as regression works reasonably well. When running

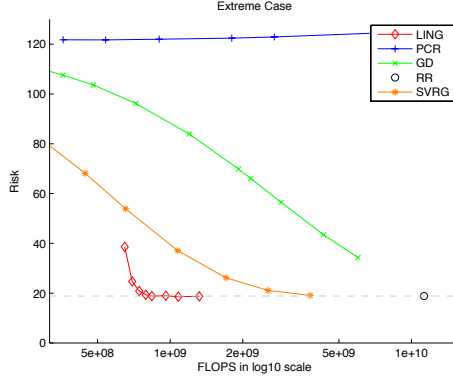


Figure 3: Model 3, Risk VS. Computational Cost plot. LING approaches RR risk the fastest. SVRG is slightly slower than LING. GD also approaches RR risk but cost more than LING. PCR has a huge risk no matter how many PCs are selected.

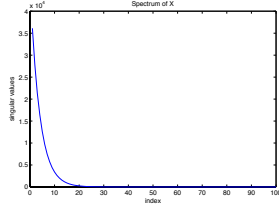


Figure 4: Top 100 singular values of \mathbf{X} in Model 1

PCR, we pick top $k_1 = 10, 20, 40, 80, 150, 300, 400$ PCs and in GD we iterate $n_1 = 2, 5, 10, 15, 20, 30, 50, 100, 150$ times. For SVRG we try $n_{\text{SVRG}} = 1, 2, 3, 5, 10, 20, 40, 80$ passes through the data. For LING we pick $k_2 = 5, 15$ PCs in the first stage and try $n_2 = 1, 2, 4, 8, 10, 15, 20, 30, 50$ iterations in the second stage. The computational cost and average classification error of the four algorithms and also the RR solution (2) on test set over 6 different train test splits are shown in figure 7. The top 150 singular values of \mathbf{X} are shown in figure 9. As shown in the figure, SVRG gets close to the RR error very fast. The two curves of LING with $k_2 = 5, 15$ are slower than SVRG since some initial FLOPS are spent on computing top PCs but after that they approach RR error very fast. GD also converges to RR but cost more than the previous two algorithms. PCR performs worst in terms of error and computational cost.

4.2.2 BUZZ IN SOCIAL MEDIA

The second dataset is the UCI buzz in social media dataset which is a regression task. The goal is to predict popularity (evaluated by the mean number of active discussion) of a certain topic on Twitter over a period. The original feature matrix contains some statistics about this topic over that period like number of discussions created and new authors interacting at the topic. The original feature dimen-

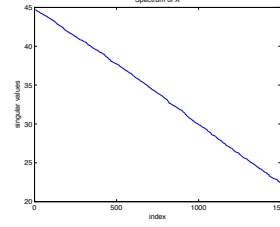


Figure 5: Singular values of \mathbf{X} in Model 2

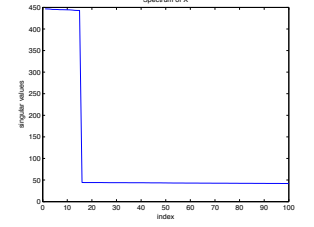


Figure 6: Top 100 singular values of \mathbf{X} in Model 3

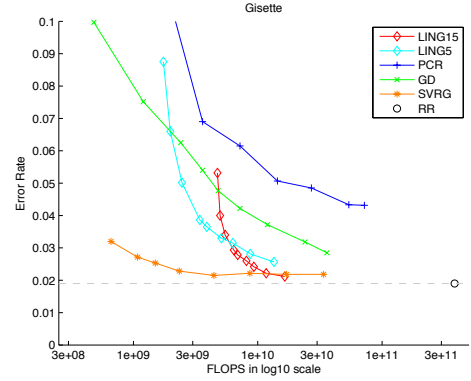


Figure 7: Gisette, Error Rate VS. Computational Cost plot. SVRG achieves small error rate fastest. Two LING lines with different n_2 spent some FLOPS on computing top PCs first, but then converges very fast to a lower error rate. GD and PCR also provide reasonably small error rate and are faster than RR, but suboptimal compared with SVRG and LING.

sion is 77. We add quadratic interactions to make it 3080. To save time, we only used a subset of 8000 samples. The samples are split into 6000 train and 2000 test. We use MSE on the test data set as the error measure. For PCR we pick $k_1 = 10, 20, 30, 50, 100, 150$ PCs and in GD we iterate $n_1 = 1, 2, 4, 6, 8, 10, 15, 20, 30, 40, 60, 100$ times. For SVRG we try $n_{\text{SVRG}} = 1, 2, 3, 5, 10, 15, 20, 40, 80$ passes through the dataset and for LING we pick $k_2 = 5, 15$ in the first stage and $n_2 = 0, 1, 2, 4, 6, 8, 10, 15, 20, 25$ iterations in the second stage. The computational cost and average MSE on test set over 5 different train test splits are shown in figure 8. The top 150 singular values of \mathbf{X} are shown in figure 10. As shown in the figure, SVRG approaches MSE of RR very fast. LING spent some initial FLOPS for computing top PCs but after that converges fast. GD and PCR also achieves reasonable performance but suboptimal compared with SVRG and LING. The MSE of PCR first decays when we add more PCs into regression but finally goes up due to overfit.

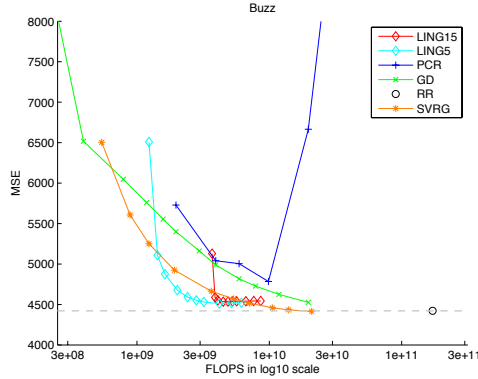


Figure 8: Buzz, MSE VS. Computational Cost plot. SVRG and two LING lines with different n_2 achieves small MSE fast. GD is slower than LING and SVRG. PCR reaches its smallest MSE at $k_1 = 50$ then overfits.

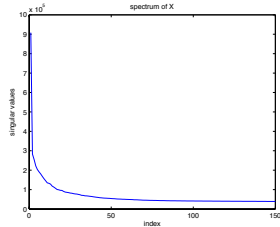


Figure 9: Top 150 singular values of \mathbf{X} in Gisette Dataset

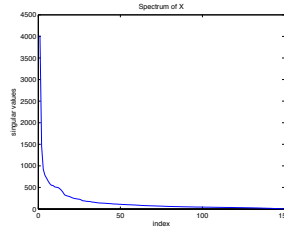


Figure 10: Top 150 singular values of \mathbf{X} in Social Media Buzz Dataset

5 SUMMARY

In this paper we present a two stage algorithm LING for computing large scale Ridge Regression which is both fast and robust in contrast to the well known approaches GD and PCR. We show that under the fixed design setting LING actually has the same risk as Ridge Regression assuming convergence. In the experiments, LING achieves good performances on all datasets when compare with three other large scale regression algorithms.

We conjecture that same strategy can be also used to accelerate the convergence of stochastic gradient descent when solving Ridge Regression since the first stage in LING essentially removes the high variance directions of \mathbf{X} , which will lead to variance reduction for the random gradient direction generated by SGD.

References

- Marina A. Epelman. Rate of convergence of steepest descent algorithm. Lecture Notes, 2007.
- Andreas Artemiou and Bing Li. On principal components and regression: a statistical explanation of a natural phe-

nomenon. *Statistica Sinica*, 19(4):1557–1565, 2009. ISSN 1017-0405.

Léon Bottou. Large-Scale Machine Learning with Stochastic Gradient Descent. In Yves Lechevallier and Gilbert Saporta, editors, *Proceedings of the 19th International Conference on Computational Statistics (COMPSTAT'2010)*, pages 177–187, Paris, France, August 2010. Springer.

Paramveer S. Dhillon, Dean P. Foster, Sham M. Kakade, and Lyle H. Ungar. A risk comparison of ordinary least squares vs ridge regression. *Journal of Machine Learning Research*, 14:1505–1511, 2013.

Isabelle Guyon, Asa Ben Hur, Steve Gunn, and Gideon Dror. Result analysis of the nips 2003 feature selection challenge. In *Advances in Neural Information Processing Systems 17*, pages 545–552. MIT Press, 2004.

N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev.*, 53(2):217–288, May 2011a. ISSN 0036-1445.

Nathan Halko, Per-Gunnar Martinsson, Yoel Shkolnisky, and Mark Tygert. An algorithm for the principal component analysis of large data sets. *SIAM J. Scientific Computing*, 33(5):2580–2594, 2011b.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in Neural Information Processing Systems (NIPS)*, 2013.

Ian Jolliffe. *Principal Component Analysis. Encyclopedia of Statistics in Behavioral Science*. John Wiley & Sons, 2005.

Yichao Lu, Paramveer S. Dhillon, Dean Foster, and Lyle Ungar. Faster ridge regression via the subsampled randomized hadamard transform. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

G. Saunders, A. Gammerman, and V. Vovk. Ridge regression learning algorithm in dual variables. In *Proc. 15th International Conf. on Machine Learning*, pages 515–521. Morgan Kaufmann, San Francisco, CA, 1998.

Tong Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *ICML 2004: Proceedings of the twenty-first International Conference on Machine Learning*. OMNIPRESS, pages 919–926, 2004.

Adaptive Monotone Shrinkage for Regression

Zhuang Ma

Department of Statistics
University of Pennsylvania
Philadelphia, PA 19104
zhuangma@wharton.upenn.edu

Dean Foster

Yahoo Lab
New York, NY 10018
dean@foster.net

Robert Stine

Department of Statistics
University of Pennsylvania
Philadelphia, PA 19104
stine@wharton.upenn.edu

Abstract

We develop an adaptive monotone shrinkage estimator for regression models with the following characteristics: i) dense coefficients with small but important effects; ii) *a priori* ordering that indicates the probable predictive importance of the features. We capture both properties with an empirical Bayes estimator that shrinks coefficients monotonically with respect to their anticipated importance. This estimator can be rapidly computed using a version of Pool-Adjacent-Violators algorithm. We show that the proposed monotone shrinkage approach is competitive with the class of all Bayesian estimators that share the prior information. We further observe that the estimator also minimizes Stein’s unbiased risk estimate. Along with our key result that the estimator mimics the oracle Bayes rule under an order assumption, we also prove that the estimator is robust. Even without the order assumption, our estimator mimics the best performance of a large family of estimators that includes the least squares estimator, constant- λ ridge estimator, James-Stein estimator, etc. All the theoretical results are non-asymptotic. Simulation results and data analysis from a model for text processing are provided to support the theory.

1 INTRODUCTION

Feature selection and coefficient estimation are familiar topics in both statistics and machine learning communities. Many results in this area concern models that are ‘nearly black,’ possessing a handful of large effects against a wide field of noise. Consider the widely used linear model

$$Y = X\beta + \epsilon \quad \text{where} \quad \epsilon \sim N(0, \sigma^2 I_n), \quad (1)$$

X is full-rank, $n \times p$ matrix of explanatory features with $p \leq n$, and β is a p dimensional vector of unknown coeffi-

cients. In the ‘nearly black’ case, all but a few of the coordinates of β are zero. A long sequence of results leverage this sparsity (Foster and George [1994]; Tibshirani [1996]; Abramovich et al. [2006]; Candes and Tao [2007]; Fan and Lv [2008]; Bickel et al. [2009]). Sparsity assumptions are well suited to many applications, especially within the field of signal and image processing (Donoho [1995], Wright et al. [2009]).

Despite the prevalence of research on sparse models, some applications do not conform to this paradigm. For example, Foster et al. [2013] used methods such as latent semantic analysis, essentially principal components analysis (PCA), to convert text into features for regression analysis. The estimated coefficients of these principal components show two specific characteristics that draw our attention: dense coefficient estimates with a monotonically decaying effect size. Rather than concentrate in a few estimates, the predictive power of the model spreads across many features. Sparsity-based methods such as hard or soft thresholding that set small effects to zero produce fitted models with greatly diminished predictive ability. Too much predictive signal has been lost by eliminating small, but nonetheless informative, coefficients. Dense coefficients appear in other applications as well. Hall et al. [2009] and Dicker [2011, 2012] also propose models for dense signals and Dicker [2011, 2012] discusses several shrinkage estimators in high dimensions.

The second characteristic of this application is the monotone decrease in typical effect size. The signal tends to concentrate in the leading principal components, then gradually decay. We may not know the signal strength, but we do have an ordering. In this sense, the unsupervised PCA of the text data provides useful information that can be exploited within the regression. In particular, the eigenvalues from the PCA provide an external ordering of the features that is suggestive of the effect size. Such exogenous information appears in other domains. In time series analysis, data collected more recently are expected to be more informative for the prediction of future trends. In principal components regression, we tend to expect the first principal

component to be more important than the second. Therefore, models without incorporating this prior knowledge might be suboptimal.

In this paper, we capture both characteristics with an empirical Bayes estimator that shrinks coefficients monotonically with respect to their anticipated importance. The procedure is tuning free and can be efficiently implemented using Pool-Adjacent-Violators algorithm. We further show that the estimator can be derived from frequentists' perspective as well by minimizing Stein's unbiased risk estimate. Finally, we establish non-asymptotic results to guarantee that the proposed estimator is nearly Bayes optimal under the order assumption and even when the order assumption (or say, prior knowledge) is wrong, it still mimics the best performance of a large family of estimators that includes the least squares estimator, ridge estimator, James-Stein estimator, etc.

The rest of this paper is organized as follows. In section 2, we describe the monotone shrinkage model in detail and introduce the maximum marginal likelihood estimator (MMLE) and Pool-Adjacent-Violators algorithm. In section 3, we show that the proposed estimator also minimizes Stein's unbiased risk estimate and establish its non-asymptotic oracle properties both with and without order assumption. In section 4, we suggest an estimator of the error variance σ^2 . In section 5, we present simulation results and an analysis of text to support our theory. Concluding remarks are given in section 6. Details of the technique are provided in the Appendix.

2 ADAPTIVE MONOTONE SHRINKAGE

2.1 Model Formulation

We use a Bayesian framework to encode the prior knowledge about the importance of explanatory features in our model. We express this prior knowledge in a distribution of the coefficients β . Intuitively, if the features within the regression are standardized such that $X^T X = I_p$, then the coefficient $|\beta_i|$ gives the importance of the i th feature: a unit change in X_i is associated with a change of $|\beta_i|$ in the response. A natural prior that captures the sense that the elements of β have decaying size specifies a monotone decreasing sequence of variances for the coefficients. For convenience, we assume that the size of signal in β_i is decreasing with the index i :

$$\beta_i \sim N(0, \sigma_i^2) \quad (2)$$

$$\sigma_1^2 \geq \sigma_2^2 \geq \dots \geq \sigma_p^2 \geq 0$$

Since we know the order, we can always rearrange β_i so that the unknown σ_i^2 are monotone as above. Throughout, we consider only orthonormal designs for which $X^T X =$

I_p . Then the Bayes rule β^* in (1) and (2) is:

$$\beta_i^* = \frac{\sigma_i^2}{\sigma_i^2 + \sigma^2} \tilde{\beta}_i,$$

where $\tilde{\beta} = (\tilde{\beta}_1, \dots, \tilde{\beta}_p) = X^T Y$ is the least squares estimator. The Bayes rule shrinks $\tilde{\beta}_i$ monotonically, shrinking more and more harshly as the index and σ_i^2 increase. For our application, we know only the order of the features, not the signal strength σ_i^2 , so the Bayes rule is not a real estimator because it depends on the unknown parameter σ_i^2 . To mimic the performance of the Bayes rule, we estimate the σ_i^2 s from data under the order constraint and use the resulting plug-in estimator. For convenience, we write the model as $\beta \sim N(0, \Sigma)$ where $\Sigma = \text{diag}(\sigma_1^2, \dots, \sigma_p^2)$.

2.2 Maximum Marginal Likelihood Estimator

To estimate the ordered prior variances on the diagonal of Σ , we observe that the marginal distribution of Y is $N(0, X \Sigma X^T + \sigma^2 I_n)$. If we further assume the error variance σ^2 from (1) is known (or we can plug in a consistent estimator), a simple calculation shows that the least square estimator $\tilde{\beta} = X^T Y$ is a sufficient statistic for Σ . So, in the following discussions, we base our inference on $\tilde{\beta}$, whose marginal distribution is $N(0, \sigma^2 I_p + \Sigma)$. A natural estimator of Σ is the maximum marginal likelihood estimator (MMLE). The log marginal likelihood function is

$$l(\Sigma) = -\frac{1}{2} \sum_{i=1}^p \left(\log(2\pi) + \log(\sigma^2 + \sigma_i^2) + \frac{\tilde{\beta}_i^2}{\sigma^2 + \sigma_i^2} \right)$$

Consequently, the MMLE is the solution to the following optimization problem:

$$\arg \min_{\sigma_i^2} \sum_{i=1}^p \left(\log(\sigma^2 + \sigma_i^2) + \frac{\tilde{\beta}_i^2}{\sigma^2 + \sigma_i^2} \right) \quad (3)$$

subject to $\sigma_1^2 \geq \sigma_2^2 \geq \dots \geq \sigma_p^2 \geq 0$

2.3 Pool-Adjacent-Violators Algorithm

The optimization problem (3) resembles the well-known isotonic regression problem

$$\hat{\beta}^{iso} = \arg \min_{\beta} \sum_{i=1}^n (y_i - \beta_i)^2 \quad \text{subject to } \beta_1 \geq \dots \geq \beta_n \quad (4)$$

whose unique solution can be efficiently obtained by running the Pool-Adjacent-Violators (PAV) algorithm. Roughly speaking, this algorithm solves (4) as follows. Set $i = 1$. Move to the right (increase the index i) until finding a pair (y_i, y_{i+1}) that violates the monotonicity constraint, that is $y_i < y_{i+1}$. Pool y_i and the adjacent y_{i+1} and replace both by their average. Next check whether $y_{i-1} < \frac{y_i + y_{i+1}}{2}$. If so, replace (y_{i-1}, y_i, y_{i+1}) with their average. Continue

to the left until monotonicity is satisfied and then proceed to the right until the whole sequence is monotone. Hence, PAV algorithm outputs an decreasing blockwise constant sequence. As far as we know, the PAV algorithm dates back to Ayer et al. [1955], where it is used to compute the MLE of independent binomial distributions. Brunk [1955, 1958] considered rather general scenarios and established some consistency properties. According to Grotzinger and Witzgall [1984], if carefully implemented, the PAV algorithm has computational complexity $O(n)$.

Although the optimization problem (3) is not convex, it can be solved efficiently by the PAV algorithm. Before establishing this result, we introduce some notations.

$$f_i(x) = \log(x + \sigma^2) + \frac{\tilde{\beta}_i^2}{x + \sigma^2}$$

$$\hat{\sigma}_i^2 = \arg \min_x f_i(x) = \tilde{\beta}_i^2 - \sigma^2$$

Proposition 2.1. *The following two-step algorithm produces the MMLE denoted by $(\hat{\sigma}_1^2, \dots, \hat{\sigma}_p^2)$*
Step 1. $(\tilde{\sigma}_1^2, \dots, \tilde{\sigma}_p^2) = \text{PAV}(\tilde{\sigma}_1^2, \dots, \tilde{\sigma}_p^2)$
Step 2. $\hat{\sigma}_i^2 = \tilde{\sigma}_i^2 I_{(\tilde{\sigma}_i^2 \geq 0)}$.

For those σ_i^2 estimated to be 0, it means the corresponding features are not included in the model. To prove the proposition, we first introduce a lemma:

Lemma 2.1. *Consider optimization problem*

$$\min \sum_{i=1}^p f_i(\theta_i)$$

subject to: $\theta_1 \geq \theta_2 \geq \dots \geq \theta_p$

where the element-by-element solution $\tilde{\theta}_i = \arg \min_{\theta} f_i(\theta)$ is finite. If the following two conditions are satisfied, then the optimization problem has the unique solution $(\hat{\theta}_1, \dots, \hat{\theta}_p) = \text{PAV}(\tilde{\theta}_1, \dots, \tilde{\theta}_p)$.

Condition 1.(Pooling Property)

Let $\bar{\theta}_{ij} = \frac{\sum_{k=i}^j \tilde{\theta}_k}{j-i+1}$. Then $\forall i \leq j$, $\arg \min_{\theta} \sum_{k=i}^j f_k(\theta) = \bar{\theta}_{ij}$ and $\sum_{k=i}^j f_k(\theta)$ is strictly decreasing when $\theta \leq \bar{\theta}_{ij}$ and strictly increasing when $\theta \geq \bar{\theta}_{ij}$.

Condition 2.(Violating Property)

If $\tilde{\theta}_i \leq \tilde{\theta}_{i+1}$, then $\hat{\theta}_i^k = \hat{\theta}_{i+1}^k, \forall i+1 \leq k \leq p$, where $(\hat{\theta}_1^k, \dots, \hat{\theta}_p^k)$ is the solution to the following optimization problem (P^k) :

$$\min \sum_{i=1}^k f_i(\theta_i)$$

subject to: $\theta_1 \geq \theta_2 \geq \dots \geq \theta_k$

Although the conditions in the lemma seem weird, actually they are nearly necessary. Helpfully, the conditions can be

easily checked. Numerous distributions have log likelihood functions that satisfy the conditions. These include the binomial distribution, Poisson distribution, normal distribution with fixed variance and variable mean, normal distribution with fixed mean and variable variance, and so on.

Proof of Proposition 2.1

The situation we are faced up with is normal distribution with fixed mean and variable variance, which satisfies the conditions in Lemma 1. According to the lemma, $(\tilde{\sigma}_1^2, \dots, \tilde{\sigma}_p^2) = \text{PAV}(\tilde{\sigma}_1^2, \dots, \tilde{\sigma}_p^2)$ solves:

$$\min \sum_{i=1}^p (\log(\sigma^2 + \sigma_i^2) + \frac{\tilde{\beta}_i^2}{\sigma^2 + \sigma_i^2})$$

subject to: $\sigma_1^2 \geq \sigma_2^2 \geq \dots \geq \sigma_p^2$

which is only slightly different from our original optimization problem. To finish the proof, we just need to introduce some auxiliary functions. Let $f_{-k} = \log(\sigma^2 + \sigma_{-k}^2) + \frac{\sigma^2}{\sigma^2 + \sigma_{-k}^2}, 1 \leq k \leq n$, so $\tilde{\sigma}_{-k}^2 = \arg \min f_{-k}(\sigma_{-k}^2) = 0$. Consider the following optimization problem (Q^n) :

$$\min \sum_{i=1}^p (\log(\sigma^2 + \sigma_i^2) + \frac{\tilde{\beta}_i^2}{\sigma^2 + \sigma_i^2})$$

$$+ \sum_{i=1}^n (\log(\sigma^2 + \sigma_{-i}^2) + \frac{\sigma^2}{\sigma^2 + \sigma_{-i}^2})$$

subject to $\sigma_1^2 \geq \dots \geq \sigma_p^2 \geq \sigma_{-1}^2 \geq \dots \geq \sigma_{-n}^2$

Lemma 1 shows $\text{PAV}(\tilde{\sigma}_1^2, \dots, \tilde{\sigma}_p^2, 0, \dots, 0)$ solves (Q^n) . Denote it $(\hat{\sigma}_{n1}^2, \dots, \hat{\sigma}_{np}^2, \hat{\sigma}_{-n1}^2, \dots, \hat{\sigma}_{-n1}^2)$. Notice that $(\hat{\sigma}_1^2, \dots, \hat{\sigma}_p^2, 0, \dots, 0)$ is a feasible solution, which should be suboptimal, that is to say,

$$\sum_{i=1}^p (\log(\sigma^2 + \hat{\sigma}_{ni}^2) + \frac{\tilde{\beta}_i^2}{\sigma^2 + \hat{\sigma}_{ni}^2}) + \sum_{i=1}^n f_{-k}(\hat{\sigma}_{-ni}^2)$$

$$\leq \sum_{i=1}^p (\log(\sigma^2 + \hat{\sigma}_i^2) + \frac{\tilde{\beta}_i^2}{\sigma^2 + \hat{\sigma}_i^2}) + \sum_{i=1}^n f_{-k}(0)$$

Recall that $\tilde{\sigma}_{-k}^2 = \arg \min f_{-k}(\sigma_{-k}^2) = 0$, which implies

$$\sum_{i=1}^p (\log(\sigma^2 + \hat{\sigma}_{ni}^2) + \frac{\tilde{\beta}_i^2}{\sigma^2 + \hat{\sigma}_{ni}^2})$$

$$\leq \sum_{i=1}^p (\log(\sigma^2 + \hat{\sigma}_i^2) + \frac{\tilde{\beta}_i^2}{\sigma^2 + \hat{\sigma}_i^2})$$

Let n goes to infinity, $\text{PAV}(\tilde{\sigma}_1^2, \dots, \tilde{\sigma}_p^2, 0, \dots, 0)$ converges to $(\tilde{\sigma}_1^2 I_{(\tilde{\sigma}_1^2 \geq 0)}, \dots, \tilde{\sigma}_p^2 I_{(\tilde{\sigma}_p^2 \geq 0)}, 0, \dots, 0)$ and the inequality above implies $(\tilde{\sigma}_1^2 I_{(\tilde{\sigma}_1^2 \geq 0)}, \dots, \tilde{\sigma}_p^2 I_{(\tilde{\sigma}_p^2 \geq 0)})$ is the solution to the original optimization problem. \square

2.4 Data-Driven Blockwise James-Stein Estimator: Global and Local Adaptivity

Proposition 1 and the nature of Pool-Adjacent-Violators algorithm show that the MMLE $(\hat{\sigma}_1^2, \dots, \hat{\sigma}_p^2)$ is decreasing and blockwise constant. We now change notation in this part and let $\hat{\sigma}_i^2$ denote the common variance estimate of the i th block. Define $\beta_i = (\beta_{i1}, \dots, \beta_{in_i})$ to be the coefficients within the i th block and correspondingly define $\tilde{\beta}_i, \tilde{\beta}_i$. With these notations, we can write explicitly $\hat{\sigma}_i^2 = \left(\frac{\sum_{j=1}^{n_i} (\tilde{\beta}_{ij}^2 - \sigma^2)}{n_i} \right)_+$ and

$$\begin{aligned} \hat{\beta}_i &= \frac{\hat{\sigma}_i^2}{\hat{\sigma}_i^2 + \sigma^2} \tilde{\beta}_i \\ &= \left(1 - \frac{n_i \sigma^2}{\sum_{j=1}^{n_i} \tilde{\beta}_{ij}^2} \right)_+ \tilde{\beta}_i, \end{aligned}$$

which is exactly the positive part of the James-Stein type estimator. Hence the proposed estimator can be interpreted as a monotone blockwise James-Stein estimator. Blockwise James-Stein estimator is well-studied in the wavelet setting (Cai [1999], Cai and Zhou [2009]). In Cai [1999], the block size is fixed before observing the data and is the same for all blocks. Cai and Zhou [2009] proposed an adaptive procedure to make the block size data-driven but the block size remains the same for all blocks. As for our procedure, the number of blocks and the size of each block are completely data-driven. The difference is due to different assumptions. The former is based on smoothness of Besov bodies while the later is based on monotonicity.

The advantage of our data-driven, monotone blockwise James-Stein estimator is the ability to achieve both global and local adaptivity. Blockwise shrinkage utilizes information about neighboring coefficients. However, if the block size is too large, local inhomogeneity might be overlooked. So, the best way to achieve a good balance is to let the data speak for itself.

3 ORACLE RISK PROPERTIES

3.1 Equivalence between MMLE and SURE Estimator

In this section, we show that our empirical Bayes estimator can also be derived within a frequentist framework by minimizing Stein's unbiased risk estimate (SURE). Under squared error loss: $l(\hat{\beta}, \beta) = \frac{1}{p} \sum_{i=1}^p (\hat{\beta}_i - \beta_i)^2$. If one uses the shrinkage estimator $\hat{\beta}^\lambda$ defined by $\hat{\beta}_i^\lambda = \frac{\lambda_i}{\lambda_i + \sigma^2} \tilde{\beta}_i$ to estimate β_i , the risk for a given β is:

$$R_p(\hat{\beta}^\lambda, \beta) = E[l(\hat{\beta}^\lambda, \beta)] = \frac{1}{p} \sum_{i=1}^p \frac{\sigma^2}{(\sigma^2 + \lambda_i)^2} (\sigma^2 \beta_i^2 + \lambda_i^2)$$

and an unbiased estimate for the risk is

$$SURE(\lambda) = \frac{1}{p} \sum_{i=1}^p \left[\left(\frac{\sigma^2}{\sigma^2 + \lambda_i} \right)^2 \tilde{\beta}_i^2 + \frac{\sigma^2 (\lambda_i - \sigma^2)}{\sigma^2 + \lambda_i} \right]$$

Generally, $SURE(\lambda)$ is unbiased estimate of the risk only if λ is a fixed constant and cannot depend on data. We say $\hat{\beta}^\lambda$ is a **monotone shrinkage estimator** if $\hat{\beta}_i^\lambda = \frac{\lambda_i}{\lambda_i + \sigma^2} \tilde{\beta}_i$ and $\hat{\lambda}_1 \geq \dots \geq \hat{\lambda}_p \geq 0$, where $\hat{\lambda}_i$ can be data dependent. A monotone shrinkage estimator is completely determined by the **monotone shrinkage parameter** $\hat{\lambda} = (\hat{\lambda}_1, \dots, \hat{\lambda}_p)$. If only considering the family of monotone shrinkage estimators, the relationship suggests that the data-dependent $\hat{\lambda}$ which minimizes $SURE(\lambda)$ should be a good choice. Define:

$$\hat{\lambda}_{SURE} = \arg \min_{\lambda_1 \geq \dots \geq \lambda_p \geq 0} \sum_{i=1}^n \left[\left(\frac{\sigma^2}{\sigma^2 + \lambda_i} \right)^2 \tilde{\beta}_i^2 + \frac{\sigma^2 (\lambda_i - \sigma^2)}{\sigma^2 + \lambda_i} \right]$$

which is of the same form as optimization problem (3). Let $g_i(\lambda_i) = \left(\frac{\sigma^2}{\sigma^2 + \lambda_i} \right)^2 \tilde{\beta}_i^2 + \frac{\sigma^2 (\lambda_i - \sigma^2)}{\sigma^2 + \lambda_i}$. Then it is easy to see that $\hat{\lambda}_i = \arg \min_{\lambda_i} g_i(\lambda_i) = \tilde{\beta}_i^2 - \sigma^2$. Checking that $g_i(\lambda_i)$ satisfy the two conditions in Lemma 2.1, the same argument used to show Proposition 2.1 implies:

Proposition 3.1. *MMLE equals SURE estimator $\hat{\beta}^{\hat{\lambda}_{SURE}}$.*

In the rest of the paper, we will use $\hat{\beta}_{SURE} = \hat{\beta}^{\hat{\lambda}_{SURE}}$ to denote the proposed estimator.

Remark 3.1. *Monotone shrinkage estimator was also investigated in Xie et al. [2012] when dealing with heteroscedastic normal sequence model. Different empirical Bayes estimators were studied in this paper and SURE estimator was shown to dominate MMLE and method of moments. While in our context, the three estimators turned out to be the same.*

3.2 Oracle Property with Order Assumption

Proposition 3.1 provides us with a powerful tool to investigate the risk properties of the proposed estimate. First of all, we introduce the oracle estimator, namely the Bayes rule $\beta^* = (\beta_1^*, \dots, \beta_p^*)$ defined by

$$\beta_i^* = \frac{\sigma_i^2}{\sigma_i^2 + \sigma^2} \tilde{\beta}_i$$

Of course, β^* is not an practical estimator because it depends on the unknown parameter $\lambda^* = (\sigma_1^2, \dots, \sigma_p^2)$. It is easy to see the oracle risk is:

$$R(\beta^*) = \frac{1}{p} \sum_{i=1}^p \frac{\sigma^2 \sigma_i^2}{\sigma^2 + \sigma_i^2}$$

Then we introduce another lemma, which is the building block of the oracle properties. It says that $E[SURE(\hat{\lambda})]$

is uniformly good approximation of the true risk $E[l(\hat{\beta}_{\hat{\lambda}})]$, where the expectation is with respect to both data and parameter β .

Lemma 3.1.

$$\sup_{\sigma_1^2, \dots, \sigma_p^2} \sup_{\hat{\lambda}_1 \geq \dots \geq \hat{\lambda}_p} |E\{E[l(\hat{\beta}_{\hat{\lambda}}, \beta) - SURE(\hat{\lambda})|\beta]\}| \leq 4\sqrt{\frac{2}{p}}\sigma^2$$

where $\hat{\lambda} = (\hat{\lambda}_1, \dots, \hat{\lambda}_p)$ is arbitrary monotone shrinkage parameter and can be data dependent.

Theorem 3.1.

$$\sup_{\sigma_1^2 \geq \dots \geq \sigma_p^2 \geq 0} (R(\hat{\beta}_{SURE}) - R(\beta^*)) \leq 4\sqrt{\frac{2}{p}}\sigma^2$$

Proof: Because λ^* is fixed constant, we have

$$\begin{aligned} R(\hat{\beta}_{SURE}, \beta) - R(\beta^*) &= E[l(\hat{\beta}_{SURE}, \beta)|\beta] - E[SURE(\lambda^*)|\beta] \\ &= E[l(\hat{\beta}_{SURE}, \beta) - SURE(\hat{\lambda}_{SURE}) + SURE(\hat{\lambda}_{SURE}) - SURE(\lambda^*)|\beta] \\ &\leq E[l(\hat{\beta}_{SURE}, \beta) - SURE(\hat{\lambda}_{SURE})|\beta] \end{aligned}$$

The inequality is due to the definition of $\hat{\lambda}_{SURE}$. So the Bayes risk satisfies:

$$\begin{aligned} R(\hat{\beta}_{SURE}) - R(\beta^*) &\leq E\{E[l(\hat{\beta}_{SURE}, \beta) - SURE(\hat{\lambda}_{SURE})|\beta]\} \\ &\leq \sup_{\sigma_1^2, \dots, \sigma_p^2} \sup_{\hat{\lambda}_1 \geq \dots \geq \hat{\lambda}_p \geq 0} |E\{E[l(\hat{\beta}_{\hat{\lambda}}, \beta) - SURE(\hat{\lambda})|\beta]\}| \end{aligned}$$

Applying lemma 3.1 finishes the proof. \square

Remark 3.2. Theorem 3.1 shows that SURE estimator mimics the oracle Bayes rule and therefore outperforms all other estimators. What needs to be highlighted is that this is a non-asymptotic result with rate of convergence $O(p^{-\frac{1}{2}})$ independent of the true σ_i^2 s. No matter how the σ_i^2 s vary, as long as the order is known, the proposed adaptive procedure can uniformly capture the truth.

Corollary 3.1. $\sup_{\sigma_1^2 \geq \dots \geq \sigma_p^2 \geq 0} \frac{R(\hat{\beta}_{SURE})}{\sigma^2 + R(\beta^*)} = 1 + 4\sqrt{\frac{2}{p}}\sigma^2$

3.3 Oracle Property without Order Assumption

In this section, we show that even without knowing the order of the σ_i^2 s, the proposed estimator retains an oracle property among monotone shrinkage estimators.

Theorem 3.2.

$$\sup_{\sigma_1^2, \dots, \sigma_p^2} (R(\hat{\beta}_{SURE}) - \inf_{\hat{\gamma}_1 \geq \dots \geq \hat{\gamma}_p \geq 0} R(\hat{\beta}_{\hat{\gamma}})) \leq 8\sqrt{\frac{2}{p}}\sigma^2$$

Proof: For any given $(\sigma_1^2, \dots, \sigma_p^2)$, we can always find $\hat{\eta} = (\hat{\eta}_1, \dots, \hat{\eta}_p)$ that satisfies $\hat{\eta}_1 \geq \dots \geq \hat{\eta}_p \geq 0$ and $R(\hat{\beta}_{\hat{\eta}}) < \inf_{\hat{\gamma}_1 \geq \dots \geq \hat{\gamma}_p \geq 0} R(\hat{\beta}_{\hat{\gamma}}) + \epsilon$. Then,

$$R(\hat{\beta}_{SURE}) - \inf_{\hat{\gamma}_1 \geq \dots \geq \hat{\gamma}_p \geq 0} R(\hat{\beta}_{\hat{\gamma}}) \leq R(\hat{\beta}_{SURE}) - R(\hat{\beta}_{\hat{\eta}}) + \epsilon$$

Notice that,

$$\begin{aligned} l(\beta, \hat{\beta}_{SURE}) - l(\beta, \hat{\beta}_{\hat{\eta}}) &= (l(\beta, \hat{\beta}_{SURE}) - SURE(\hat{\lambda}_{SURE})) \\ &+ (SURE(\hat{\lambda}_{SURE}) - SURE(\hat{\eta})) + (SURE(\hat{\eta}) - l(\beta, \hat{\beta}_{\hat{\eta}})) \\ &\leq (l(\beta, \hat{\beta}_{SURE}) - SURE(\hat{\lambda}_{SURE})) + (SURE(\hat{\eta}) - l(\beta, \hat{\beta}_{\hat{\eta}})) \end{aligned}$$

Take expectations, we have

$$\begin{aligned} R(\hat{\beta}_{SURE}) - R(\hat{\beta}_{\hat{\eta}}) &\leq E\{E[l(\beta, \hat{\beta}_{SURE}) - SURE(\hat{\lambda}_{SURE}) + SURE(\hat{\eta}) - l(\beta, \hat{\beta}_{\hat{\eta}})|\beta]\} \\ &\leq 2 \sup_{\sigma_1^2, \dots, \sigma_p^2} \sup_{\hat{\lambda}_1 \geq \dots \geq \hat{\lambda}_p \geq 0} |E\{E[l(\beta_{\hat{\lambda}}, \beta) - SURE(\hat{\lambda})|\beta]\}| \end{aligned}$$

Lemma 3.1 implies,

$$R(\hat{\beta}_{SURE}) - \inf_{\hat{\gamma}_1 \geq \dots \geq \hat{\gamma}_p \geq 0} R(\hat{\beta}_{\hat{\gamma}}) \leq 8\sqrt{\frac{2}{p}}\sigma^2 + \epsilon$$

Since the upper bound does not depend on σ_i^2 , let $\epsilon \rightarrow 0$, and the theorem follows. \square

If we replace the data dependent shrinkage parameters in Theorem 2 with fixed ones, we can improve the error bound by a factor of 2, which is

Corollary 3.2.

$$\sup_{\sigma_1^2, \dots, \sigma_p^2} (R(\hat{\beta}_{SURE}) - \inf_{\hat{\gamma}_1 \geq \dots \geq \hat{\gamma}_p \geq 0} R(\hat{\beta}_{\hat{\gamma}})) \leq 4\sqrt{\frac{2}{p}}\sigma^2$$

Theorem 2 shows that even when the order assumption is invalid, the proposed estimator is nearly the best in the family of monotone shrinkage estimators. In particular, uniform shrinkage estimators such as least square estimator, ridge estimator and James-Stein estimator and stepwise regression methods such as monotone AIC, BIC, RIC (just search for p nested submodels: with i th submodel as $\{1, \dots, i\}$) are included. This is also a non-asymptotic result with rate of convergence $O(p^{-\frac{1}{2}})$ independent of the σ_i^2 s. Therefore, the proposed estimator is robust and good enough for practical use. Actually, Theorem 3.2 states about the worst case. If the order is partially right, the proposed procedure benefits where the order is right and retains good properties where the order is wrong.

Remark 3.3. The robustness is due to the ‘soft constraint’. Instead of restricting the norm of regression coefficients to be monotone, we incorporate the constraint in the prior distribution, which makes the model flexible and robust.

4 ESTIMATION OF σ^2

We have assumed σ^2 is known to establish the theoretical properties of our estimator. Here we suggest a reasonable estimate in practice that is based on maximum marginal likelihood. Unlike section 2.2, within this section the unknown parameter becomes $\theta = (\sigma_1^2, \dots, \sigma_p^2, \sigma^2)$. Recall that the marginal distribution of Y is $N(0, X\Sigma X^T + \sigma^2 I_n)$. So, the log marginal likelihood function:

$$l(\theta|y) \propto -\log(|X\Sigma X^T + \sigma^2 I_n|) - y^T (X\Sigma X^T + \sigma^2 I_n)^{-1} y$$

where $|\cdot|$ means determinant. Let $X = (x_1, \dots, x_p)$, we can add another $n - p$ vectors x_{p+1}, \dots, x_n to make $\tilde{X} = (X, x_{p+1}, \dots, x_n)$ an orthonormal matrix. Let $\tilde{\Sigma} = \text{diag}(\Sigma + \sigma^2 I_p, \sigma^2 I_{n-p})$. Then $X\Sigma X^T + \sigma^2 I_n = \tilde{X}\tilde{\Sigma}\tilde{X}^T$ and thus $(X\Sigma X^T + \sigma^2 I_n)^{-1} = \tilde{X}\tilde{\Sigma}^{-1}\tilde{X}^T$. Plug this expression back into the marginal likelihood function,

$$l(\theta|y) \propto -\log(|\tilde{X}\tilde{\Sigma}\tilde{X}^T|) - y^T \tilde{X}\tilde{\Sigma}^{-1}\tilde{X}^T y$$

We abuse notation in this section and let $\tilde{\beta} = \tilde{X}^T y$. If we introduce variable $(\tau_1^2, \dots, \tau_n^2) = \text{diag}(\tilde{\Sigma}) = (\sigma_1^2 + \sigma^2, \dots, \sigma_p^2 + \sigma^2, \sigma^2, \dots, \sigma^2)$, then

$$l(\Sigma) \propto -\sum_{i=1}^n (\log \tau_i^2 + \frac{\tilde{\beta}_i^2}{\tau_i^2})$$

So, the MMLE is the solution to the following optimization problem.

$$\min \sum_{i=1}^p (\log \tau_i^2 + \frac{\tilde{\beta}_i^2}{\tau_i^2})$$

$$\text{subject to: } \tau_1^2 \geq \dots \geq \tau_p^2 \geq \tau_{p+1}^2 = \dots = \tau_n^2 \geq 0$$

Following a similar but slightly different argument in Proposition 2.1, we get:

Proposition 4.1. *The solution is uniquely given by*

$$(\hat{\tau}_1^2, \dots, \hat{\tau}_p^2, \hat{\tau}_{p+1}^2, \dots, \hat{\tau}_n^2) = \text{PAV}(\tilde{\beta}_1^2, \dots, \tilde{\beta}_p^2, \frac{\sum_{i=p+1}^n \tilde{\beta}_i^2}{n-p}, \dots, \frac{\sum_{i=p+1}^n \tilde{\beta}_i^2}{n-p})$$

The MMLE of original parameters can be recovered by $(\hat{\tau}_1^2, \dots, \hat{\tau}_p^2, \hat{\tau}_{p+1}^2, \dots, \hat{\tau}_n^2) = (\hat{\sigma}_1^2 + \hat{\sigma}^2, \dots, \hat{\sigma}_p^2 + \hat{\sigma}^2, \hat{\sigma}^2, \dots, \hat{\sigma}^2)$.

5 NUMERICAL EXPERIMENTS

5.1 Simulation Results

In this section, we compare the proposed monotone shrinkage approach with several other popular methods for feature selection and estimation. For simplicity, we only consider the normal sequence model and assume the error variance σ^2 is known.

- PAV, the proposed adaptive monotone shrinkage procedure computed by Pool-Adjacent-Violators algorithm.
- Lasso, with λ selected by minimizing Stein's unbiased risk estimate. Under orthogonal design, it is also known as Sureshrink (Donoho and Johnstone [1995]).
- Ridge estimator with λ selected by Cross-Validation.
- Positive part of James-Stein estimator
- Classical stepwise regression, we use AIC for penalty criterion.
- Monotone AIC: AIC that just searches for p nested submodels, i.e., with k_{th} submodel $= \{1, \dots, k\}$

We consider the following scenarios ($p = 100, \sigma^2 = 1$):

1. Signals with Decaying Size: $(\sigma_1^2, \dots, \sigma_p^2)$ are generated from decreasing order statistics of $2\chi^2$.
2. Signals with Same Size: $\sigma_i^2 = 2, \forall 1 \leq i \leq p$
3. Sparse Signals: first 90% of the σ_i^2 are 0 and remaining 10% of σ_i^2 are generated from decreasing order statistics of $4\chi^2$.
4. Signals with Increasing Size: $(\sigma_1^2, \dots, \sigma_p^2)$ are generated from increasing order statistics of $2\chi^2$. This scenario does not satisfy our order assumption (actually, the worst case), which is used to show the robustness of our procedure.

With $(\sigma_1^2, \dots, \sigma_p^2)$ fixed, we adopt the following simulation strategy.

1. Generate $\beta = (\beta_1, \dots, \beta_p)$ by $\beta_i \sim N(0, \sigma_i^2)$
2. Condition on β , generate the observation $X = (x_1, \dots, x_p)$ by $x_i \sim N(\beta_i, \sigma^2)$
3. Use the methods discussed above to estimate the signal β and compute the mean square error.
4. Repeat 1-3 for 400 times. The average of the mean square errors is an estimate of the Bayes risk.

Mean square error condition on different β are given below using box plot and the middle line of each box represents Bayes risk of each procedure. The red line in the figure stands for the oracle risk, i.e., the Bayes risk of the oracle Bayes rule.

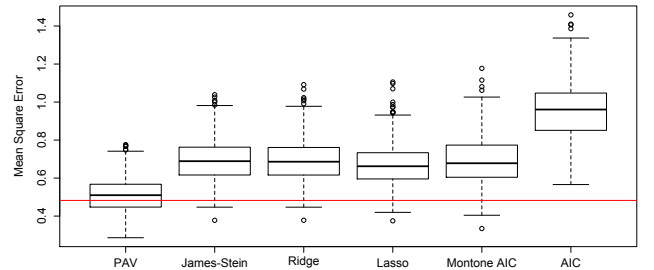


Figure 1: Signals with decaying size, i.e. $\{\sigma_i^2\}$ is decreasing. The adaptive monotone shrinkage procedure pools signals of similar sizes together and shrinks blockwisely and monotonically. Red line stands for oracle Bayes risk.

For signals with decaying size, oracle estimator shrink monotonically with respect to the size of the signals. Uniform shrinkage estimators such as ridge and James-Stein estimator are suboptimal. The proposed adaptive monotone procedure makes use of the prior information and mimics the oracle Bayes rule by pooling signals of similar size together so that it shrinks blockwisely and monotonically. AIC overfits the data while monotone AIC makes use of the order structure and therefore performs better.

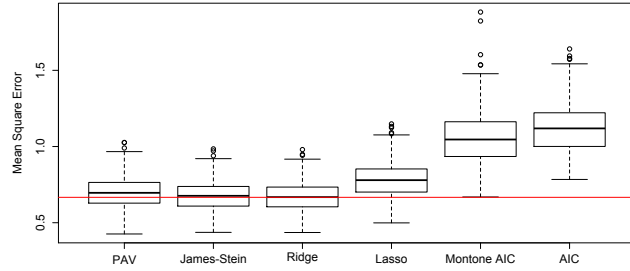


Figure 2: Signals with same size, i.e. σ_i^2 s are the same. This is the case where ridge and James-Stein estimator capture the truth with full power while our procedure will regard the σ_i^2 s as different(decreasing) and will generally divide the σ_i^2 s into more than one blocks, which leads to slight power loss. Red line stands for oracle Bayes risk.

For signals with same size, the oracle estimator shrink uniformly. Ridge and James-Stein estimator mimic the oracle Bayes rule with full power. The proposed adaptive procedure does not necessarily guarantee uniform shrinkage but the power loss is negligible.

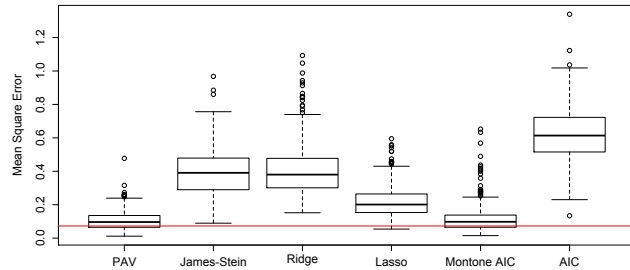


Figure 3: Sparse Signals, i.e. the size of the signals remain decreasing while 90% of them are 0. The proposed monotone shrinkage procedure can effectively kill the noise and shrink the signals properly. Red line stands for oracle Bayes risk.

For sparse signals, the oracle estimator kill the noise and shrink the signals monotonically. Monotone AIC can efficiently distinguish signal and noise while does not shrink the signals. The proposed adaptive procedure not only kills the noise but also shrinks the real signals properly according to their sizes. For those methods that cannot make use of the order structure, Lasso does better in this sparse case.

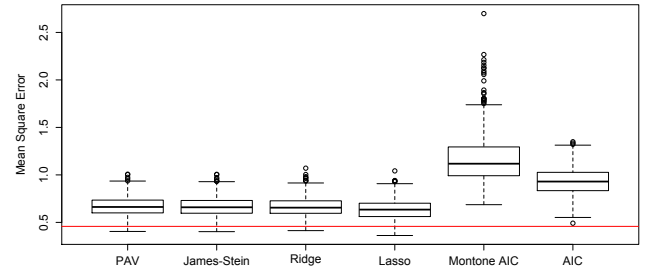


Figure 4: Signals with increasing size, i.e. $\{\sigma_i^2\}$ is increasing, which is opposite to our assumption that the size of signals is decaying. The adaptive monotone shrinkage procedure is robust and performs as good as other estimators. Red line stands for oracle Bayes risk.

For signals with increasing size, the proposed estimator uses completely reverse order. As theorem 2 expects, wrong prior knowledge won't ruin our estimator. It still mimics the best performance of the monotone shrinkage family. However, monotone AIC, which is not as robust as our procedure, suffers a lot from wrong prior knowledge.

5.2 Analysis of Text Processing Data

In this section, we apply the proposed adaptive monotone shrinkage approach to text data of real estate described in Foster et al. [2013]. The features included in the regression model are the leading 1500 principal components of the bag-of-words of text. The response is the log transformation of the real estate price. We use the eigenvalues from PCA to order the effect size of the features (see Figure 5 for the absolute t-statistics of the leading 500 principal components). Although the data dose not ideally satisfy the assumptions of our model, the proposed adaptive procedure is robust enough to leverage this rough prior knowledge.

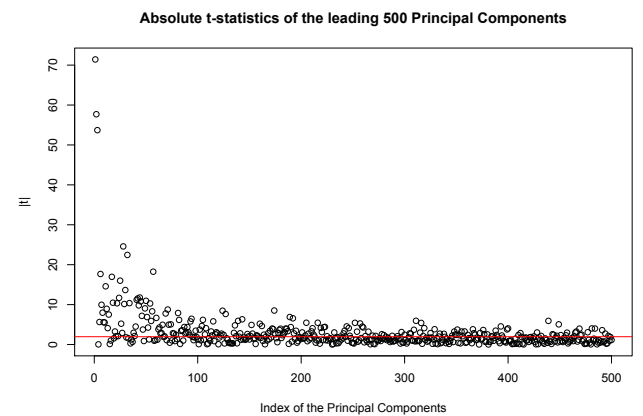


Figure 5: Absolute t-statistics of the leading 500 Principal Components. Those above the red line are significant.

The sample size is 7384 and we use 10 fold cross validation to estimate the prediction error of each procedure.

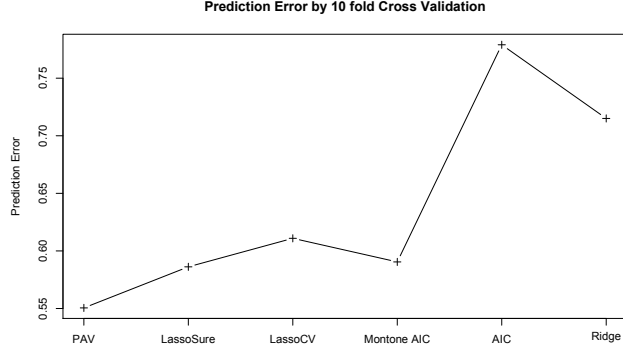


Figure 6: Prediction error comparison of different methods. LassoSURE: Lasso with tuning paramter selected by minimizing Stein's unbiased risk estimate. LassoCV: Lasso computed by LARS (Efron et al. [2004]) and paramter tuned by cross validation.

The result shows

- PAV outperforms Ridge regression. From Figure 5, we can see that the signals are of different sizes. Uniform shrinkage method shrink the important features too much while shrink weak signals less harshly than it should be. PAV can adaptively pool the signals of similar size together and shrink blockwisely and monotonically.
- PAV outperforms LassoSure and LassoCV. Lasso can capture the sparse pattern of the data but as sacrifice, it might shrink important features a bit more than they should be.
- PAV outperforms Monotone AIC. Both procedures make use of prior information but PAV is more robust. There are several informative principal components corresponding to small eigenvalues so that Monotone AIC will exclude them from the model.

6 CONCLUSIONS

In this paper, we proposed an adaptive monotone shrinkage approach for regression with features of ordered effect size. We showed that the procedure can be rapidly computed via Pool-Adjacent-Violators algorithm and holds oracle risk properties. Non-asymptotic results are established. Furthermore, although the procedure is based on knowing the right prior knowledge about the features, we proved that, when the prior knowledge is wrong or in the absence of prior knowledge, the estimator still mimics the best performance of the family of monotone shrinkage estimators. Hence, it is robust enough to use in practice.

Compared with penalized least square methods which require heavy computational effort to find the best regularization paramter, the proposed adaptive procedure is tuning free. As noticed in the analysis of text data, the monotone shrinkage approach naturally works with PCA since the

principal components are essentially ordered and orthogonal. Recent devolopments in randomized algorithms(Halko et al. [2011]) enable us to quickly compute the PCA of a huge matrix so that the proposed procedure can be easily applied to large-scale datasets.

7 APPENDIX

7.1 Proof of Lemma 2.1

It is sufficient to prove the following two claims:

- i) $(\hat{\theta}_1^k, \dots, \hat{\theta}_k^k) = \text{PAV}(\tilde{\theta}_1, \dots, \tilde{\theta}_k), 1 \leq k \leq p$ ii) $\forall 1 \leq i, j \leq k, \hat{\theta}_i^k = \hat{\theta}_j^k \Rightarrow \hat{\theta}_i^m = \hat{\theta}_j^m, \forall k \leq m \leq p$

We prove the claim by induction:

1. It is trivial for $k = 1$ since $\hat{\theta}_1^1 = \tilde{\theta}_1$
2. Assuming the claim holds for k . If $\tilde{\theta}_{k+1} < \hat{\theta}_k^k$, we can see that

$$\begin{aligned}
 & \sum_{i=1}^k f_i(\hat{\theta}_i^k) + f_{k+1}(\tilde{\theta}_{k+1}) \\
 &= \min_{\theta_1 \geq \dots \geq \theta_k} \sum_{i=1}^k f_i(\theta_i) + \min_{\theta_{k+1}} f_{k+1}(\theta_{k+1}) \\
 &\leq \min_{\theta_1 \geq \dots \geq \theta_{k+1}} \sum_{i=1}^{k+1} f_i(\theta_i)
 \end{aligned}$$

which implies,

$$\begin{aligned}
 (\hat{\theta}_1^{k+1}, \dots, \hat{\theta}_{k+1}^{k+1}) &= (\hat{\theta}_1^k, \dots, \hat{\theta}_k^k, \tilde{\theta}_{k+1}) \\
 &= \text{PAV}(\tilde{\theta}_1, \dots, \tilde{\theta}_k)
 \end{aligned}$$

So we prove claim i). Notice that $\hat{\theta}_{k+1}^{k+1} \neq \hat{\theta}_j^{k+1}, \forall j \leq k$, claim ii) is true by induction.

If $\tilde{\theta}_{k+1} \geq \hat{\theta}_k^k$, denote j the smallest integer such that $\hat{\theta}_j^k = \hat{\theta}_{j+1}^k = \dots = \hat{\theta}_k^k$. Because the boundary condition is not active between θ_{j-1} and θ_j , we can conclude that $(\hat{\theta}_j^k, \hat{\theta}_{j+1}^k, \dots, \hat{\theta}_k^k) = \arg \min_{\theta_j \geq \dots \geq \theta_k} \sum_{i=j}^k f_i(\theta_i)$. Then

condition 1 implies that $\hat{\theta}_j^k = \dots = \hat{\theta}_k^k = \frac{\sum_{i=j}^k \tilde{\theta}_i}{k-j+1} \leq \tilde{\theta}_{k+1}$.

We claim: $\hat{\theta}_j^m = \hat{\theta}_{j+1}^m = \dots = \hat{\theta}_{k+1}^m, \forall k \leq m \leq p$. By induction we have already known that $\hat{\theta}_j^m = \hat{\theta}_{j+1}^m = \dots = \hat{\theta}_k^m$. If $\hat{\theta}_k^m \leq \tilde{\theta}_{k+1}$, then by condition 1, $f_{k+1}(\theta_{k+1})$ is strictly decreasing on $(0, \hat{\theta}_k^m)$, which forces $\hat{\theta}_{k+1}^m = \hat{\theta}_k^m$.

If $\hat{\theta}_k^m \geq \tilde{\theta}_{k+1}$, then $\hat{\theta}_{k+1}^m \geq \tilde{\theta}_{k+1}$. $\tilde{\theta}_{k+1} \geq \frac{\sum_{i=j}^k \tilde{\theta}_i}{k-j+1}$ and condition 1 force $\hat{\theta}_k^m = \hat{\theta}_{k+1}^m$. Thus we proved $\hat{\theta}_j^m = \hat{\theta}_{j+1}^m = \dots = \hat{\theta}_{k+1}^m, \forall m > k$. Specifically, $\hat{\theta}_j^{k+1} = \hat{\theta}_{j+1}^{k+1} = \dots = \hat{\theta}_{k+1}^{k+1}$. If $\frac{\sum_{i=j}^{k+1} \tilde{\theta}_i}{k-j} \leq \hat{\theta}_{j-1}^k$, again by condition 1, $\hat{\theta}_j^{k+1} = \dots = \hat{\theta}_{k+1}^{k+1} = \frac{\sum_{i=j}^{k+1} \tilde{\theta}_i}{k-j}$ and consequently $\hat{\theta}_i^{k+1} = \hat{\theta}_i^k, 1 \leq i \leq j-1$. We are done because the

solution is exactly PAV($\tilde{\theta}_1, \dots, \tilde{\theta}_{k+1}$), which proves claim i) and $\hat{\theta}_j^m = \hat{\theta}_{j+1}^m = \dots = \hat{\theta}_{k+1}^m, \forall m > k$ implies claim ii). If $\frac{\sum_{i=j}^{k+1} \tilde{\theta}_i}{k-j} > \hat{\theta}_{j-1}^k$, assume i to be the smallest integer such that $\hat{\theta}_i^k = \hat{\theta}_{i+1}^k = \dots = \hat{\theta}_{j-1}^k$. By similar argument, we can prove that $\hat{\theta}_i^m = \hat{\theta}_{i+1}^m = \dots = \hat{\theta}_{k+1}^m, \forall m > k$. If $\frac{\sum_{t=i}^{k+1} \tilde{\theta}_t}{k-i} < \hat{\theta}_{i-1}^k$, we are done. If not, continue the same argument. \square

7.2 Proof of Lemma 3.1

Plug in the expression of SURE(λ), we have

$$\begin{aligned} & E[l(\beta_{\hat{\lambda}}, \beta) - \text{SURE}(\hat{\lambda})|\beta] \\ &= E\left[\frac{1}{p} \sum_{i=1}^p \frac{2\hat{\lambda}_i}{\sigma^2 + \hat{\lambda}_i} (\tilde{\beta}_i^2 - \tilde{\beta}_i \beta_i - \sigma^2)\right] \\ & - (\tilde{\beta}_i^2 - \sigma^2 - \beta_i^2)|\beta] = E\left[\frac{1}{p} \sum_{i=1}^p \frac{2\hat{\lambda}_i}{\sigma^2 + \hat{\lambda}_i} (\tilde{\beta}_i^2 - \tilde{\beta}_i \beta_i - \sigma^2)|\beta\right] \end{aligned}$$

Take expectation with respect to β , we get

$$\begin{aligned} & E\{E[l(\beta_{\hat{\lambda}}, \beta) - \text{SURE}(\hat{\lambda})|\beta]\} = \\ & E\left\{E\left[\frac{1}{p} \sum_{i=1}^p \frac{2\hat{\lambda}_i}{\sigma^2 + \hat{\lambda}_i} (\tilde{\beta}_i^2 - \tilde{\beta}_i \beta_i - \sigma^2)|\beta\right]\right\} \end{aligned}$$

Notice that $\beta_i|\tilde{\beta}_i \sim N(\frac{\sigma_i^2}{\sigma^2 + \sigma_i^2} \tilde{\beta}_i, \frac{\sigma^2 \sigma_i^2}{\sigma^2 + \sigma_i^2})$ and the marginal distribution of $\tilde{\beta}_i$ is $N(0, \sigma^2 + \sigma_i^2)$, we change the order of expectation and get:

$$\begin{aligned} & E\{E[l(\beta_{\hat{\lambda}}, \beta) - \text{SURE}(\hat{\lambda})|\beta]\} = \\ & 2E\left[\frac{1}{p} \sum_{i=1}^p \frac{\hat{\lambda}_i}{\sigma^2 + \hat{\lambda}_i} \left(\frac{\sigma^2}{\sigma^2 + \sigma_i^2} \tilde{\beta}_i^2 - \sigma^2\right)\right] \end{aligned}$$

where the expectation is with respect to $\tilde{\beta}_i \sim N(0, \sigma^2 + \sigma_i^2)$.

$$\begin{aligned} & |E\{E[l(\beta_{\hat{\lambda}}, \beta) - \text{SURE}(\hat{\lambda})|\beta]\}| \leq \\ & 2\sigma^2 E\left|\frac{1}{p} \sum_{i=1}^p \frac{\hat{\lambda}_i}{\sigma^2 + \hat{\lambda}_i} \left(\frac{\tilde{\beta}_i^2}{\sigma^2 + \sigma_i^2} - 1\right)\right| \\ & \leq 2\sigma^2 E\left\{\sup_{1 \leq c_1 \leq \dots \leq c_p \geq 0} \frac{1}{p} \left|\sum_{i=1}^p c_i \left(\frac{\tilde{\beta}_i^2}{\sigma^2 + \sigma_i^2} - 1\right)\right|\right\} \\ & = 2\sigma^2 E\left\{\sup_{1 \leq c_1 \leq \dots \leq c_p \geq 0} \frac{1}{p} \left|\sum_{i=1}^p c_i (Z_i - 1)\right|\right\} \end{aligned}$$

where $Z_i \sim i.i.d \chi^2$. Observe that

$$\begin{aligned} & \sup_{1 \leq c_1 \leq \dots \leq c_p \geq 0} \left|\frac{1}{p} \sum_{i=1}^p c_i (Z_i - 1)\right| \\ &= \max_{1 \leq j \leq p} \left|\frac{1}{p} \sum_{i=1}^j (Z_i - 1)\right| \end{aligned}$$

which is also used in Lemma 7.2 of Li [1985] and Theorem 3.1 in Xie et al. [2012], we have:

$$\begin{aligned} & |E\{E[l(\beta_{\hat{\lambda}}, \beta) - \text{SURE}(\hat{\lambda})|\beta]\}| \\ & \leq 2\sigma^2 E\left\{\max_{1 \leq j \leq p} \left|\frac{1}{p} \sum_{i=1}^j (Z_i - 1)\right|\right\} \end{aligned}$$

Let $M_j = \sum_{i=1}^j (Z_i - 1)$, then M_j is a martingale. So the L^2 maximal inequality implies:

$$E\left(\max_{1 \leq j \leq p} M_j^2\right) \leq 4E(M_p^2) = 8p$$

$$\begin{aligned} & |E\{E[l(\beta_{\hat{\lambda}}, \beta) - \text{SURE}(\hat{\lambda})|\beta]\}| \\ & \leq 2\sigma^2 E\left\{\max_{1 \leq j \leq p} \left|\frac{1}{p} \sum_{i=1}^j (Z_i - 1)\right|\right\} \\ & \leq \frac{2\sigma^2}{p} (E(\max_{1 \leq j \leq p} M_j)^2)^{\frac{1}{2}} \end{aligned}$$

Combine the two inequalities, we have

$$|E\{E[l(\beta_{\hat{\lambda}}, \beta) - \text{SURE}(\hat{\lambda})|\beta]\}| \leq 4\sqrt{\frac{2}{p}}\sigma^2$$

Since the error bound does not depend on $\hat{\lambda}$ and σ_i^2 , the lemma follows. \square

References

- Felix Abramovich, Yoav Benjamini, David L Donoho, and Iain M Johnstone. Adapting to unknown sparsity by controlling the false discovery rate. *The Annals of Statistics*, 34(2):584–653, 2006.
- Miriam Ayer, H Daniel Brunk, George M Ewing, WT Reid, and Edward Silverman. An empirical distribution function for sampling with incomplete information. *The Annals of Mathematical Statistics*, pages 641–647, 1955.
- Peter J Bickel, Ya'acov Ritov, and Alexandre B Tsybakov. Simultaneous analysis of lasso and dantzig selector. *The Annals of Statistics*, pages 1705–1732, 2009.
- H. D. Brunk. Maximum likelihood estimates of monotone parameters. *The Annals of Mathematical Statistics*, 26(4):607–616, 1955.
- H. D. Brunk. On the estimation of parameters restricted by inequalities. *The Annals of Mathematical Statistics*, pages 437–454, 1958.
- T Tony Cai. Adaptive wavelet estimation: a block thresholding and oracle inequality approach. *Annals of statistics*, pages 898–924, 1999.
- T Tony Cai and Harrison H Zhou. A data-driven block thresholding approach to wavelet estimation. *The Annals of Statistics*, pages 569–595, 2009.

- Emmanuel Candes and Terence Tao. The dantzig selector: statistical estimation when p is much larger than n . *The Annals of Statistics*, pages 2313–2351, 2007.
- Lee Dicker. Dense signals, linear estimators, and out-of-sample prediction for high-dimensional linear models. *arXiv preprint arXiv:1102.2952*, 2011.
- Lee Dicker. Optimal estimation and prediction for dense signals in high-dimensional linear models. *arXiv preprint arXiv:1203.4572*, 2012.
- David L Donoho. De-noising by soft-thresholding. *Information Theory, IEEE Transactions on*, 41(3):613–627, 1995.
- David L Donoho and Iain M Johnstone. Adapting to unknown smoothness via wavelet shrinkage. *Journal of the american statistical association*, 90(432):1200–1224, 1995.
- Bradley Efron, Trevor Hastie, Iain Johnstone, Robert Tibshirani, et al. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004.
- Jianqing Fan and Jinchi Lv. Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(5):849–911, 2008.
- Dean P Foster and Edward I George. The risk inflation criterion for multiple regression. *The Annals of Statistics*, pages 1947–1975, 1994.
- Dean P Foster, Mark Liberman, and Robert A Stine. Featurizing text: Converting text into predictors for regression analysis. 2013.
- SJ Grotzinger and C Witzgall. Projections onto order simplexes. *Applied mathematics and optimization*, 12(1): 247–270, 1984.
- Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.
- Peter Hall, Jiashun Jin, and Hugh Miller. Feature selection when there are many influential features. *arXiv preprint arXiv:0911.4076*, 2009.
- Ker-Chau Li. From stein’s unbiased risk estimates to the method of generalized cross validation. *The Annals of Statistics*, pages 1352–1377, 1985.
- Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.
- John Wright, Allen Y Yang, Arvind Ganesh, Shankar S Sastry, and Yi Ma. Robust face recognition via sparse representation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(2):210–227, 2009.
- Xianchao Xie, SC Kou, and Lawrence D Brown. Sure estimates for a heteroscedastic hierarchical model. *Journal of the American Statistical Association*, 107(500):1465–1479, 2012.

Firefly Monte Carlo: Exact MCMC with Subsets of Data

Dougal Maclaurin
Department of Physics
Harvard University
Cambridge, MA 02138

Ryan P. Adams
School of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138

Abstract

Markov chain Monte Carlo (MCMC) is a popular and successful general-purpose tool for Bayesian inference. However, MCMC cannot be practically applied to large data sets because of the prohibitive cost of evaluating every likelihood term at every iteration. Here we present *Firefly Monte Carlo* (FlyMC) an auxiliary variable MCMC algorithm that only queries the likelihoods of a potentially small subset of the data at each iteration yet simulates from the exact posterior distribution, in contrast to recent proposals that are approximate even in the asymptotic limit. FlyMC is compatible with a wide variety of modern MCMC algorithms, and only requires a lower bound on the per-datum likelihood factors. In experiments, we find that FlyMC generates samples from the posterior more than an order of magnitude faster than regular MCMC, opening up MCMC methods to larger datasets than were previously considered feasible.

1 INTRODUCTION

The Bayesian approach to probabilistic modeling is appealing for a several reasons: the generative framework allows one to separate out modeling assumptions from inference procedures, outputs include estimates of uncertainty that can be used for decision making and prediction, and it provides clear ways to perform model selection and complexity control. Unfortunately, the fully-Bayesian approach to modeling is often very computationally challenging. It is unusual for non-trivial models of real data to have closed-form posterior distributions. Instead, one uses approximate inference via Monte Carlo, variational approximations, Laplace approximations, or other tools.

One of the persistent challenges to Bayesian computation is that coherent procedures for inference appear to require

examination of all of the data in order to evaluate a new hypothesis regarding parameters or latent variables. For example, when performing Metropolis–Hastings (MH), it is necessary to evaluate the target posterior density for each proposed parameter update, and this posterior will usually contain a factor for each datum. Similarly, typical variational Bayesian procedures need to build local approximations for each of the data in order to update the approximation to any global parameters. In both cases, it may be necessary to perform these data-intensive computations many times as part of an iterative procedure.

Recent methods have been proposed to partially overcome these difficulties. Stochastic and online variational approximation procedures (Hoffman et al., 2010, 2013) can use subsets of the data to make approximations to global parameters. As these procedures are optimizations, it is possible to build on convergence results from the stochastic optimization literature and achieve guarantees on the resulting approximation. For Markov chain Monte Carlo, the situation is somewhat murkier. Recent work has shown that approximate transition operators based on subsets of data can be used for predictive prefetching to help parallelize MCMC (Angelino et al., 2014). Other work uses approximate transition operators directly, for Metropolis–Hastings (MH) and related algorithms (Welling and Teh, 2011). Korattikara et al. (2014) and Bardenet et al. (2014) have shown that such approximate MH moves can lead to stationary distributions which are approximate but that have bounded error, albeit under strong conditions of rapid mixing.

In this paper, we present a Markov chain Monte Carlo algorithm, *Firefly Monte Carlo* (FlyMC), that is in line with these latter efforts to exploit subsets of data to construct transition operators. What distinguishes the approach we present here, however, is that this new MCMC procedure is *exact* in the sense that it leaves the true full-data posterior distribution invariant. FlyMC is a latent variable model which introduces a collection of Bernoulli variables – one for each datum – with conditional distributions chosen so that they effectively turn on and off data points in the posterior, hence “firefly”. The introduction of these latent vari-

ables does not alter the marginal distribution of the parameters of interest. Our only requirement is that it be possible to provide a “collapsible” lower bound for each likelihood term. FlyMC can lead to dramatic performance improvements in MCMC, as measured in wallclock time.

The paper is structured as follows. In Section 2, we introduce Firefly Monte Carlo and show why it is valid. Section 3 discusses practical issues related to implementation of FlyMC. Section 4 evaluates the new method on several different problems, and Section 5 discusses its limitations and possible future directions.

2 FIREFLY MONTE CARLO

The Firefly Monte Carlo algorithm tackles the problem of sampling from the posterior distribution of a probabilistic model. We will denote the parameters of interest as θ and assume that they have prior $p(\theta)$. We assume that N data have been observed $\{x_n\}_{n=1}^N$ and that these data are conditionally independent given θ under a likelihood $p(x_n | \theta)$. Our target distribution is therefore

$$p(\theta | \{x_n\}_{n=1}^N) \propto p(\theta, \{x_n\}_{n=1}^N) = p(\theta) \prod_{n=1}^N p(x_n | \theta). \quad (1)$$

For notational convenience, we will write the n th likelihood term as a function of θ as

$$L_n(\theta) = p(x_n | \theta).$$

An MCMC sampler makes transitions from a given θ to a new θ' such that posterior distribution remains invariant. Conventional algorithms, such as Metropolis–Hastings, require evaluation of the unnormalized posterior in full at every iteration. When the data set is large, evaluating all N likelihoods is a computational bottleneck. This is the problem that we seek to solve with FlyMC.

For each data point, n , we introduce a binary auxiliary variable, $z_n \in \{0, 1\}$, and a function $B_n(\theta)$ which is a strictly positive lower bound on the n th likelihood: $0 < B_n(\theta) \leq L_n(\theta)$. Each z_n has the following Bernoulli distribution conditioned on the parameters:

$$p(z_n | x_n, \theta) = \left[\frac{L_n(\theta) - B_n(\theta)}{L_n(\theta)} \right]^{z_n} \left[\frac{B_n(\theta)}{L_n(\theta)} \right]^{1-z_n}.$$

We now augment the posterior distribution with these N variables:

$$\begin{aligned} p(\theta, \{z_n\}_{n=1}^N | \{x_n\}_{n=1}^N) &\propto p(\theta, \{x_n, z_n\}_{n=1}^N) \\ &= p(\theta) \prod_{n=1}^N p(x_n | \theta) p(z_n | x_n, \theta). \end{aligned}$$

As in other auxiliary variable methods such as slice sampling, Swendsen-Wang, or Hamiltonian Monte Carlo, augmenting the joint distribution in this way does not damage

the original marginal distribution of interest:

$$\begin{aligned} &\sum_{z_1} \cdots \sum_{z_N} p(\theta) \prod_{n=1}^N p(x_n | \theta) p(z_n | x_n, \theta) \\ &= p(\theta) \prod_{n=1}^N p(x_n | \theta) \sum_{z_n} p(z_n | x_n, \theta) \\ &= p(\theta) \prod_{n=1}^N p(x_n | \theta) \end{aligned}$$

However, this joint distribution has a remarkable property: to evaluate the probability density over θ , given a particular configuration of $\{z_n\}_{n=1}^N$, it is only necessary to evaluate those likelihood terms for which $z_n = 1$. Consider factor n from the product above:

$$\begin{aligned} &p(x_n | \theta) p(z_n | x_n, \theta) \\ &= L_n(\theta) \left[\frac{L_n(\theta) - B_n(\theta)}{L_n(\theta)} \right]^{z_n} \left[\frac{B_n(\theta)}{L_n(\theta)} \right]^{1-z_n} \\ &= \begin{cases} L_n(\theta) - B_n(\theta) & \text{if } z_n = 1 \\ B_n(\theta) & \text{if } z_n = 0 \end{cases}. \end{aligned}$$

The “true” likelihood term $L_n(\theta)$ only appears in those factors for which $z_n = 1$ and we can think of these data as forming a “minibatch” subsample of the full set. If most $z_n = 0$, then transition updates for the parameters will be much cheaper, as these are applied to $p(\theta | \{x_n, z_n\}_{n=1}^N)$.

Of course, we do have to evaluate all N bounds $B_n(\theta)$ at each iteration. At first glance, we seem to have just shifted the computational burden from evaluating the $L_n(\theta)$ to evaluating the $B_n(\theta)$. However, if we choose $B_n(\theta)$ to have a convenient form, a scaled Gaussian or other exponential family distribution, for example, then the full product $\prod_{n=1}^N B_n(\theta)$ can be computed for each new θ in $O(1)$ time using the sufficient statistics of the distribution, which only need to be computed once. To make this clearer, we can rearrange the joint distribution in terms of a “pseudo-prior,” $\tilde{p}(\theta)$ and “pseudo-likelihood,” $\tilde{L}_n(\theta)$ as follows:

$$p(\theta, \{z_n\}_{n=1}^N | \{x_n\}_{n=1}^N) \propto \tilde{p}(\theta) \prod_{n: z_n=1} \tilde{L}_n(\theta) \quad (2)$$

where the product only runs over those n for which $z_n = 1$, and we have defined

$$\tilde{p}(\theta) = p(\theta) \prod_{n=1}^N B_n(\theta) \quad \tilde{L}_n(\theta) = \frac{L_n(\theta) - B_n(\theta)}{B_n(\theta)}.$$

We can generate a Markov chain for the joint distribution in Equation (2) by alternating between updates of θ conditional on $\{z_n\}_{n=1}^N$, which can be done with any conventional MCMC algorithm, and updates of $\{z_n\}_{n=1}^N$ conditional on θ for which we discuss efficient methods in Section 3.2. We emphasize that the marginal distribution over

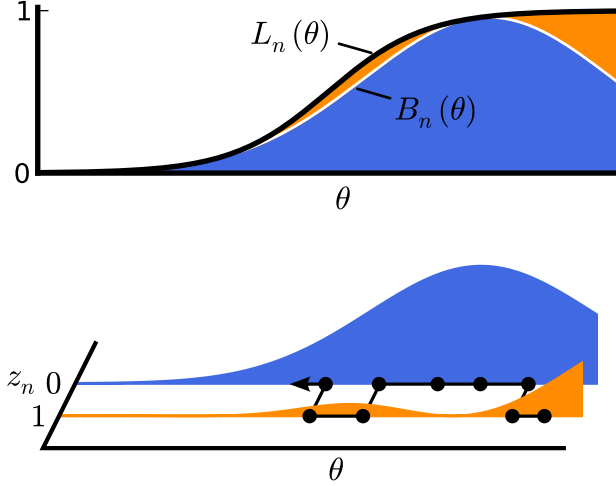


Figure 1: Illustration of the auxiliary variable representation of a single likelihood for a one-dimensional logistic regression model. The top panel shows how the likelihood function, $L_n(\theta)$, corresponding to a single datum n , can be partitioned into two parts: a lower bound, $B_n(\theta)$, shaded blue, and the remainder, shaded orange. The bottom panel shows that we can introduce a Bernoulli random variable z_n and construct a Markov chain in this new, higher dimensional space, such that marginalizing out (i.e. ignoring) the z_n recovers the original likelihood. If $B_n(\theta) \gg L_n(\theta) - B_n(\theta)$, the Markov chain will tend to occupy $z_n = 0$ and we can avoid evaluating $L_n(\theta)$ at each iteration.

θ is still the correct posterior distribution given in Equation (1).

At a given iteration, the $z_n = 0$ data points are “dark”: we simulate the Markov chain without computing their likelihoods. Upon a Markov transition in the space of $\{z_n\}_{n=1}^N$, a smattering of these dark data points become “bright” with their $z_n = 1$, and we include their likelihoods in subsequent iterations. The evolution of the chain evokes an image of fireflies, as the individual data blink on and off due to updates of the z_n .

The details of choosing a lower bound and efficiently sampling the $\{z_n\}$ are treated in the proceeding sections, but the high-level picture is now complete. Figure 1 illustrates the augmented space, and a simple version of the algorithm is shown in Algorithm 1. Figure 2 shows several steps of Firefly Monte Carlo on a toy logistic regression model.

3 IMPLEMENTATION CONSIDERATIONS

In this section we discuss two important practical matters for implementing an effective FlyMC algorithm: how to

choose and compute lower bounds, and how to sample the brightness variables z_n . For this discussion we will assume that we are dealing with a data set consisting of N data points, and a parameter set, θ , of dimension $D \ll N$. We will also assume that it takes at least $O(ND)$ time to evaluate the likelihoods at some θ for the whole data set and that evaluating this set of likelihoods at each iteration is the computational bottleneck for MCMC. We will mostly assume that space is not an issue: we can hold the full data set in memory and we can afford additional data structures occupying a few bytes for each of the N data.

The goal of an effective implementation of FlyMC is to construct a Markov chain with similar convergence and mixing properties to that of regular MCMC, while only evaluating a subset of the data points on average at each iteration. If the average number of “bright” data points is M , we would like this to achieve a computational speedup of nearly N/M over regular MCMC.

3.1 Choosing a lower bound

The lower bounds, $B_n(\theta)$ of each data point’s likelihood $L_n(\theta)$ should satisfy two properties. They should be relatively tight, and it should be possible to efficiently summarize a product of lower bounds $\prod_n B_n(\theta)$ in a way that (after setup) can be evaluated in time independent of N .

The tightness of the bounds is important because it determines the number of bright data points at each iteration, which determines the time it takes to evaluate the joint posterior. For a burned-in chain, the average number of bright data points, M , will be:

$$M = \sum_{n=1}^N \langle z_n \rangle = \sum_{n=1}^N \int p(\theta | \{x_n\}_{n=1}^N) \frac{L_n(\theta) - B_n(\theta)}{L_n(\theta)} d\theta.$$

Therefore it is important that the bounds are tight at values of θ where the posterior puts the bulk of its mass.

The second important property is that the product of the lower bounds must be easy to compute and represent. This property emerges naturally if we use scaled exponential-family lower bounds so that their product can be summarized via a set of sufficient statistics. We should also mention that the individual bounds $B_n(\theta)$ should be easy to compute themselves, since these are computed alongside $L_n(\theta)$ for all the bright points at each iteration. In all the examples considered in this paper, the rate-limiting step in computing either $L_n(\theta)$ or $B_n(\theta)$ is the evaluation of the dot product of a feature vector with a vector of weights. Once we have computed $L_n(\theta)$ the extra cost of computing $B_n(\theta)$ is negligible.

At this stage it is useful to consider a concrete example. The logistic regression likelihood is

$$L_n(\theta) = \text{logit}^{-1}(t_n \theta^\top x_n) = \frac{1}{1 + \exp\{-t_n \theta^\top x_n\}},$$

Algorithm 1 Firefly Monte Carlo*Note: Using simple random-walk MH for clarity.*

```

1:  $\theta_0 \sim \text{INITIALDIST}$  ▷ Initialize the Markov chain state.
2: for  $i \leftarrow 1 \dots \text{ITERS}$  do ▷ Iterate the Markov chain.
3:   for  $j \leftarrow 1 \dots \lceil N \times \text{RESAMPLEFRACTION} \rceil$  do
4:      $n \sim \text{RandInteger}(1, N)$  ▷ Select a random data point.
5:      $z_n \sim \text{Bernoulli}(1 - B_n(\theta_{i-1})/L_n(\theta_{i-1}))$  ▷ Biased coin-flip to determine whether  $n$  is bright or dark.
6:   end for
7:    $\theta' \leftarrow \theta_{i-1} + \eta$  where  $\eta \sim \text{Normal}(0, \epsilon^2 \mathbb{I}_D)$  ▷ Make a random walk proposal with step size  $\epsilon$ .
8:    $u \sim \text{Uniform}(0, 1)$  ▷ Draw the MH threshold.
9:   if  $\frac{\text{JOINTPOSTERIOR}(\theta'; \{z_n\}_{n=1}^N)}{\text{JOINTPOSTERIOR}(\theta; \{z_n\}_{n=1}^N)} > u$  then ▷ Evaluate MH ratio conditioned on auxiliary variables.
10:     $\theta_i \leftarrow \theta'$  ▷ Accept proposal.
11:   else
12:     $\theta_i \leftarrow \theta_{i-1}$  ▷ Reject proposal and keep current state.
13:   end if
14: end for
15:
16: function  $\text{JOINTPOSTERIOR}(\theta; \{z_n\}_{n=1}^N)$  ▷ Modified posterior that conditions on auxiliary variables.
17:    $P \leftarrow p(\theta) \times \prod_{n=1}^N B_n(\theta)$  ▷ Evaluate prior and bounds. Collapse of bound product not shown.
18:   for each  $n$  for which  $z_n = 1$  do ▷ Loop over bright data only.
19:      $P \leftarrow P \times (L_n(\theta)/B_n(\theta) - 1)$  ▷ Include bound-corrected factor.
20:   end for
21:   return  $P$ 
22: end function

```

where $x_n \in \mathbb{R}^D$ is the set of features for the n th data point and $t_n \in \{-1, 1\}$ is its class. The logistic function has a family of scaled Gaussian lower bounds, described in Jaakkola and Jordan (1997), parameterized by ξ , the location at which the bound is tight:

$$\log(B_n(\theta)) = a(t_n \theta^\top x_n)^2 + b(t_n \theta^\top x_n) + c$$

where:

$$\begin{aligned} a &= \frac{-1}{4\xi} \left(\frac{e^\xi - 1}{e^\xi + 1} \right) & b &= \frac{1}{2} \\ c &= -a * \xi^2 + \frac{\xi}{2} - \log(e^\xi + 1) \end{aligned}$$

This is the bound shown in Fig. 1. The product of these bounds can be computed for a given θ in $O(D^2)$ time, provided we have precomputed the moments of the data, at a one-time setup cost of $O(ND^2)$:

$$\frac{1}{N} \log \prod_{n=1}^N B_n(\theta) = a \theta^\top \hat{S} \theta + b \theta^\top \hat{\mu} + c$$

where

$$\hat{S} = \frac{1}{N} \sum_{n=1}^N x_n x_n^\top \quad \hat{\mu} = \frac{1}{N} \sum_{n=1}^N t_n x_n.$$

This bound can be quite tight. For example, if we choose $\xi = 1.5$ the probability of a data point being bright

is less than 0.02 in the region where $0.1 < L_n(\theta) < 0.9$. With a bit of up-front work, we can do even better than this by choosing bounds that are tight in the right places. For example, we can perform a quick optimization to find an approximate maximum *a posteriori* (MAP) value of θ and construct the bounds to be tight there. We explore this idea further in Section 4.

3.2 Sampling and handling the auxiliary brightness variables

The resampling of the z_n variables, as shown in lines 3 to 6 of Algorithm 1, takes a step by explicitly sampling z_n from its conditional distribution for a random fixed-size subset of the data. We call this approach *explicit resampling* and it has a clear drawback: if the fixed fraction is α (shown as RESAMPLEFRACTION in Algorithm 1), then the chain cannot have a mixing time faster than $1/\alpha$, as each data point is only visited a fraction of the time.

Nevertheless, explicit resampling works well in practice since the bottleneck for mixing is usually the exploration of the space of θ , not space of z_n . Explicit resampling has the benefit of being a simple, low-overhead algorithm that is easy to vectorize for speed. The variant shown in Algorithm 1 is the simplest: data points are chosen at random, with replacement. We could also sample without replacement but this is slightly harder to do efficiently. Another variant would be to deterministically choose a subset from which to Gibbs sample at each iteration. This is more in

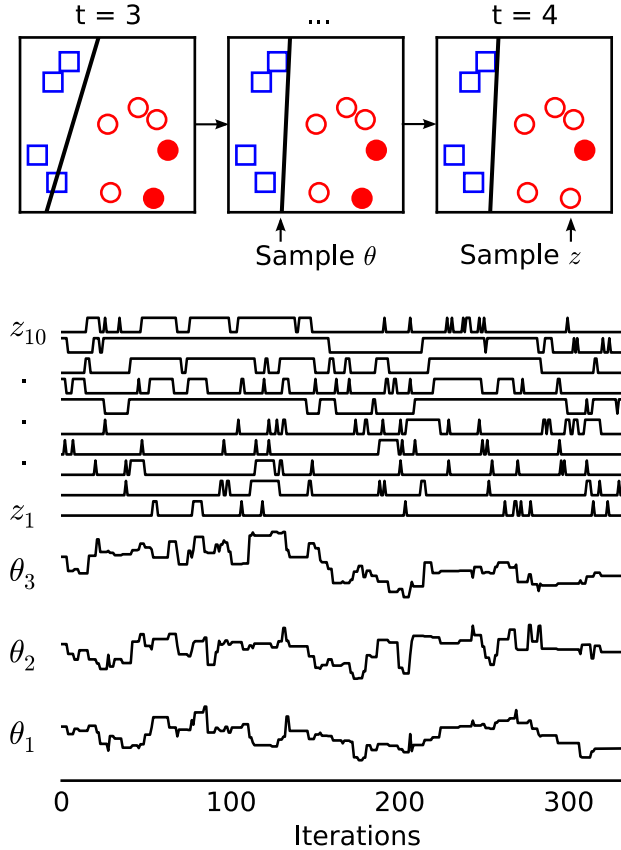


Figure 2: Illustration of the FlyMC algorithm operating on a logistic regression model of a toy synthetic data set, a two-class classification problem in two dimensions (and one bias dimension). The top panel shows a single iteration of FlyMC, from $t = 3$ to $t = 4$, which consists of two steps: first we sample θ , represented by the line of equal class probability. Next we sample the z_n . In this case, we see one ‘bright’ (solid) data point become dark. The bottom panel shows the trajectories of all components of θ and z .

line with the traditional approach of stochastic gradient descent optimization. Such an approach may be appropriate for data sets which are too large to fit into memory, since we would no longer need random access to all data points. The resulting Markov chain would be non-reversible, but still satisfy stationarity conditions.

Explicitly sampling a subset of the z_n seems wasteful if $M \ll N$, since most updates to z_n will leave it unchanged. We can do better by drawing each update for z_n from a pair of tunable Bernoulli proposal distributions $q(z'_n = 1 | z_n = 0) = q_{d \rightarrow b}$ and $q(z'_n = 0 | z_n = 1) = q_{b \rightarrow d}$, and then performing a Metropolis–Hastings accept/reject step with the true auxiliary probability $p(z_n | x_n, \theta)$. This proposal can be efficiently made for each data point, but it is only necessary

to evaluate $p(z_n | x_n, \theta)$ – and therefore the likelihood function – for the subset of data points which are proposed to change state. That is, if a sample from the proposal distribution sends $z_n = 0$ to $z_n = 0$ then it doesn’t matter whether we accept or reject. If we use samples from a geometric distribution to choose the data points, it is not even necessary to explicitly sample all of the N proposals.

The probabilities $q_{b \rightarrow d}$ and $q_{d \rightarrow b}$ can be tuned as hyper-parameters. If they are larger than $p(z_n = 0 | x_n, \theta)$ and $p(z_n = 1 | x_n, \theta)$ respectively, then we obtain near-perfect Gibbs sampling. But larger values also require more likelihood evaluations per iteration. Since the likelihoods of the bright data points have already been evaluated in the course of the Markov step in θ we can reuse these values and set $q_{b \rightarrow d} = 1$, leaving $q_{d \rightarrow b}$ as the only hyper-parameter, which we can set to something like M/N . The resulting algorithm, which we call *implicit resampling*, is shown as Algorithm 2.

3.3 Data structure for brightness variables

In the algorithms shown so far, we have aimed to construct a valid Markov chain while minimizing the number of likelihood evaluations, on the (reasonable) assumption that likelihood evaluations dominate the computational cost. However, the algorithms presented do have some steps which appear to scale linearly with N , even when M is constant. These are steps such as “loop over the bright data points” which takes time linear in N . With a well-chosen data structure for storing the variables z_n , we can ensure that these operations only scale with M .

The data structure needs to store the values of z_n for all n from 1 to N , and it needs to support the following methods in $O(1)$ time:

- `Brighten(n)`: Set $z_n = 1$
- `ithBright(i)`: Return n , the i th bright data point (in some arbitrary ordering).

We similarly require `Darken` and `ithDark`. The data structure should also keep track of how many bright data points there are.

To achieve this, we use the cache-like data structure shown in Figure 3. We store two arrays of length N . The first is `z.arr`, which contains a single copy of each of the indices n from 1 to N . All of the bright indices appear before the dark indices. A variable `z.B` keeps track of how many bright indices there are, and thus where the bright-dark transition occurs. In order to also achieve $O(1)$ assignment of indices, we also maintain a direct lookup table `z.tab` whose n th entry records the position in array `z.arr` where n is held. `Brighten(n)` works by looking up `z.tab` the position of n in `z.arr`, swapping it with

Algorithm 2 Implicit z_n sampling

```
1: for  $n \leftarrow 1 \dots N$  do                                     ▷ Loop over all the auxiliary variables.
2:   if  $z_n = 1$  then                                         ▷ If currently bright, propose going dark.
3:      $u \sim \text{Uniform}(0, 1)$                                    ▷ Sample the MH threshold.
4:     if  $\frac{q_{d \rightarrow b}}{\tilde{L}_n(\theta)} > u$  then             ▷ Compute MH ratio with  $\tilde{L}_n(\theta)$  cached from  $\theta$  update.
5:        $z_n \leftarrow 0$                                        ▷ Flip from bright to dark.
6:     end if
7:   else                                                       ▷ Already dark, consider proposing to go bright.
8:     if  $v < q_{d \rightarrow b}$  where  $v \sim \text{Uniform}(0, 1)$  then   ▷ Flip a biased coin with probability  $q_{d \rightarrow b}$ .
9:        $u \sim \text{Uniform}(0, 1)$                                    ▷ Sample the MH threshold.
10:      if  $\frac{\tilde{L}_n(\theta)}{q_{d \rightarrow b}} < u$  then                 ▷ Compute MH ratio.
11:         $z_n \leftarrow 1$                                        ▷ Flip from dark to bright.
12:      end if
13:    end if
14:  end if
15: end for
```

the index at position $z.B$, incrementing $z.B$, and updating $z.tab$ accordingly.

4 EXPERIMENTS

For FlyMC to be a useful algorithm it must be able to produce effectively independent samples from posterior distributions more quickly than regular MCMC. We certainly expect it to iterate more quickly than regular MCMC since it evaluates fewer likelihoods per iteration. But we might also expect it to mix more slowly, since it has extra auxiliary variables. To see whether this trade-off works out in FlyMC’s favor we need to know how much faster it iterates and how much slower it mixes. The answer to the first question will depend on the data set and the model. The answer to the second will depend on these too, and also on the choice of algorithm for updating θ .

We conducted three experiments, each with a different data set, model, and parameter-update algorithm, to give an impression of how well FlyMC can be expected to perform. In each experiment we compared FlyMC, with two choices of bound selection, to regular full-posterior MCMC. We looked at the average number of likelihoods queried at each iteration and the number of effective samples generated per iteration, accounting for autocorrelation. The results are summarized in Figure 4 and Table 1. The broad conclusion is that FlyMC offers a speedup of at least one order of magnitude compared with regular MCMC if the bounds are tuned according to a MAP-estimate of θ . In the following subsections we describe the experiments in detail.

4.1 Logistic regression

We applied FlyMC to the logistic regression task described in Welling and Teh (2011) using the Jaakkola-

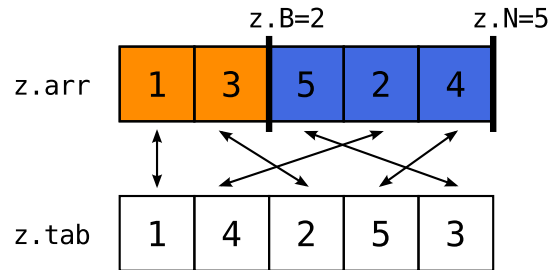


Figure 3: Illustration of a data structure allowing for efficient operations on the sets of bright and dark data points. Data points 1 and 3 are bright, the rest are dark.

Jordan bounds described earlier. The task is to classify MNIST 7s and 9s, using the first 50 principal components (and one bias) as features. We used a Gaussian prior over the weights and chose the scale of that prior by evaluating performance on a held-out test set. To sample over θ , we used symmetric Metropolis-Hasting proposals, with step size chosen to yield an acceptance rate of 0.234 (Roberts et al., 1997), optimized for each algorithm separately. We sampled the z_n using the implicit Metropolis-Hastings sampling algorithm.

We compared three different algorithms: regular MCMC, untuned FlyMC, and MAP-tuned FlyMC. For untuned FlyMC, we chose $\xi = 1.5$ for all data points. To compute the bounds for the MAP-tuned algorithm, we performed stochastic gradient descent optimization to find a set of weights close to the MAP value and gave each data point its own ξ to make the bounds tight at the MAP parameters: $L_n(\theta_{\text{MAP}}) = B_n(\theta_{\text{MAP}})$ for all n . For untuned FlyMC, and MAP-tuned FlyMC we used $q_{d \rightarrow b} = 0.1$ and $q_{d \rightarrow b} = 0.01$ respectively, chosen to be similar to the typi-

		Algorithm	Average Likelihood queries per iteration	Effective Samples per 1000 iterations	Speedup relative to regular MCMC
Data set:	MNIST	Regular MCMC	12,214	3.7	(1)
Model:	Logistic regression	Untuned FlyMC	6,252	1.3	0.7
Updates:	Metropolis-Hastings	MAP-tuned FlyMC	207	1.4	22
Data set:	3-Class CIFAR-10	Regular MCMC	18,000	8.0	(1)
Model:	Softmax classification	Untuned FlyMC	8,058	4.2	1.2
Updates:	Langevin	MAP-tuned FlyMC	654	3.3	11
Data set:	OPV	Regular MCMC	18,182,764	1.3	(1)
Model:	Robust regression	Untuned FlyMC	2,753,428	1.1	5.7
Updates:	Slice sampling	MAP-tuned FlyMC	575,528	1.2	29

Table 1: Results from empirical evaluations. Three experiments are shown: logistic regression applied to MNIST digit classification, softmax classification for three categories of CIFAR-10, and robust regression for properties of organic photovoltaic molecules, sampled with random-walk Metropolis–Hastings, Langevin-adjusted Metropolis, and slice sampling, respectively. For each of these, the vanilla MCMC operator was compared with both untuned FlyMC and FlyMC where the bound was determined from a MAP estimate of the posterior parameters. We use likelihood evaluations as an implementation-independent measure of computational cost and report the number of such evaluations per iteration, as well as the resulting sample efficiency (computed via R-CODA (Plummer et al., 2006)), and relative speedup.

cal fraction of bright data points in each case.

The results are shown in Figure 4a and summarized in Table 1. On a per-iteration basis, the FlyMC algorithms mix and burn-in more slowly than regular MCMC by around a factor of two, as illustrated by the autocorrelation plots. Even on a per-likelihood basis, the naïve FlyMC algorithm, with a fixed ξ , performs worse than regular MCMC, by a factor of 0.7, despite needing fewer likelihood evaluations per iteration. The MAP-tuned algorithm was much more impressive: after burn-in, it queried only 207 of the 12,2214 likelihoods per iteration on average, giving a speedup of more than 20, even taking into account the slower per-iteration mixing time. We initialized all chains with draws from the prior. Notice that the MAP-tuned algorithm performs poorly during burn-in, since the bounds are less tight during this time, whereas the reverse is true for the untuned algorithm.

4.2 Softmax classification

Logistic regression can be generalized to multi-class classification problems by softmax classification. The softmax likelihood of a data point belonging to class k of K classes is

$$L_n(\theta) = \frac{\exp(\theta_k^\top x_n)}{\sum_{k'=1}^K \exp(\theta_{k'}^\top x_n)}$$

Where θ is now a $K \times D$ matrix. The Jaakkola-Jordan bound does not apply to this softmax likelihood, but we can use a related bound, due to Böhning (1992), whose log matches the value and gradient of the log of the softmax likelihood at some particular θ , but has a tighter curvature. Murphy (2012) has the result in full in the chapter on variational inference.

We applied softmax classification to a three-class version of CIFAR-10 (airplane, automobile and bird) using 256 binary features discovered by Krizhevsky (2009) using a deep autoencoder. Once again, we used a Gaussian prior on the weights, chosen to maximize out-of-sample performance. This time we used the Metropolis-adjusted Langevin algorithm (MALA, Roberts and Tweedie (1996)) for our parameter updates. We chose the step sizes to yield acceptance rates close to the optimal 0.57 (Roberts and Rosenthal, 1998). Other parameters were tuned as in the logistic regression experiment.

The softmax experiment gave qualitatively similar results to the logistic regression experiment, as seen in Figure 4b and Table 1. Again, the MAP-tuned FlyMC algorithm dramatically outperformed both the lackluster untuned FlyMC and regular MCMC, offering an 11-fold speedup over the latter.

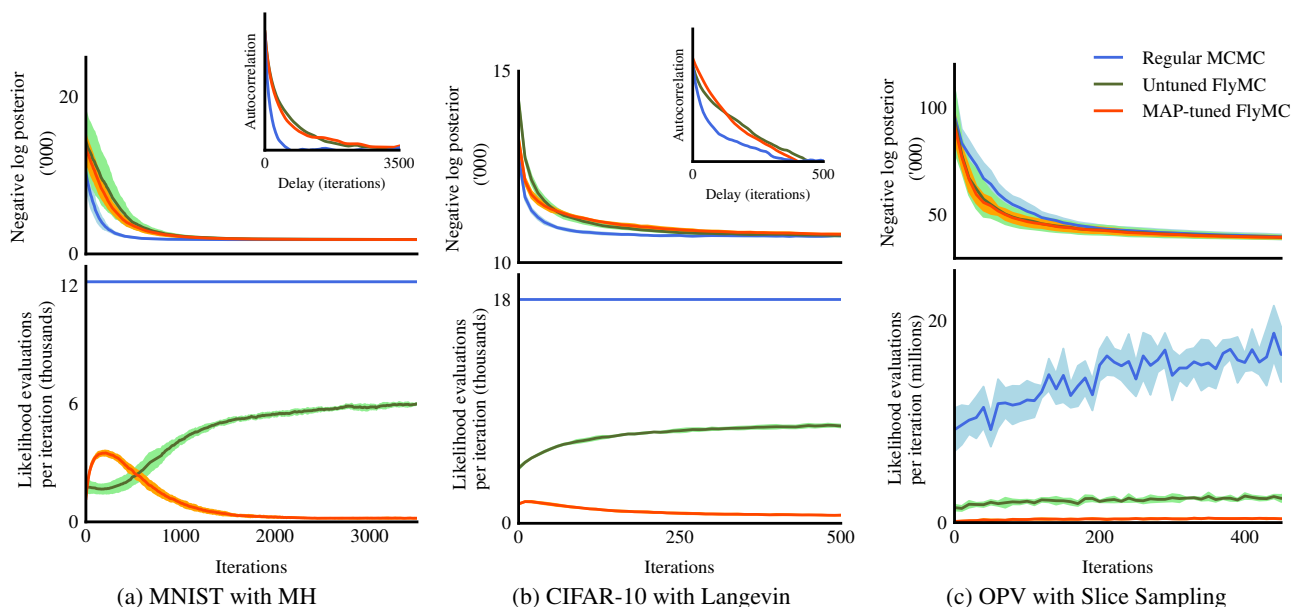


Figure 4: Tuned and untuned Firefly Monte Carlo compared to regular MCMC with three different operators, data sets, and models: (a) the digits 7 and 9 from the MNIST data are classified using logistic regression, with a random-walk Metropolis-Hastings operator; (b) softmax classification on three classes (airplane, automobile, and bird) from the CIFAR-10 image dataset, using Langevin-adjusted Metropolis; (c) robust regression on the HOMO-LUMO gap (as computed by density functional theory calculations) for a large set of organic photovoltaic molecules, using slice sampling. In each subfigure, the top shows the trace of the log posterior density to illustrate convergence, and the bottom shows the average number of likelihoods computed per iteration. One standard deviation is shown around the mean value, as computed from five runs of each. The blue lines are computed using the full-data posterior, and the green and orange lines show the untuned and tuned Firefly MC traces, respectively.

4.3 Robust sparse linear regression

Linear regression with Gaussian likelihoods yields a closed-form expression for the posterior. Non-Gaussian likelihoods, however, like heavy-tailed distributions used in so-called “robust regression” do not. Our final experiment was to perform inference over robust regression weights for a very large dataset of molecular features and computed electronic properties. The data set, described by Hachmann et al. (2011, 2014) consists of 1.8 million molecules, with 57 cheminformatic features each (Oliveres-Amaya et al., 2011; Amador-Bedolla et al., 2013). The task was to predict the HOMO-LUMO energy gap, which is useful for predicting photovoltaic efficiency.

We used a student-t distribution with $\nu = 4$ for the likelihood function and we computed a Gaussian lower bound to this by matching the value and gradient of the t distribution probability density function value at some ξ ($\xi = 0$ for the untuned case, $\xi = \theta_{MAP}^T x$ for the MAP-tuned case). We used a sparsity-inducing Laplace prior on the weights. As before, we chose the scales of the prior and the likelihood to optimize out-of sample performance.

We performed parameter updates using slice sampling (Neal, 2003). Note that slice sampling results in a variable

number of likelihood evaluations per iteration, even for the regular MCMC algorithm. Again, we found that MAP-tuned FlyMC substantially outperformed regular MCMC, as shown in Figure 4c and Table 1.

5 DISCUSSION

In this paper, we have presented Firefly Monte Carlo, an algorithm for performing Markov chain Monte Carlo using subsets (minibatches) of data. Unlike other recent proposals for such MCMC operators, FlyMC is exact in the sense that it has the true full-data posterior as its target distribution. This is achieved by introducing binary latent variables whose states represent whether a given datum is bright (used to compute the posterior) or dark (not used in posterior updates). By carefully choosing the conditional distributions of these latent variables, the true posterior is left intact under marginalization. The primary requirement for this to be efficient is that the likelihoods term must have lower bounds that collapse in an efficient way.

There are several points that warrant additional discussion and future work. First, we recognize that useful lower bounds can be difficult to obtain for many problems. It would be helpful to produce such bounds automatically for

a wider class of problems. As variational inference procedures are most often framed in terms of lower bounds on the marginal likelihood, we expect that Firefly Monte Carlo will benefit from developments in so-called “black box” variational methods (Wang and Blei, 2013; Ranganath et al., 2014). Second, we believe we have only scratched the surface of what is possible with efficient data structures and latent-variable update schemes. For example, the MH proposals we consider here for z_n have a fixed global $q_{d \rightarrow b}$, but clearly such a proposal should vary for each datum. Third, it is often the case that larger state spaces lead to slower MCMC mixing. In Firefly Monte Carlo, much like other auxiliary variable methods, we have expanded the state space significantly. We have shown empirically that the slower mixing can be more than offset by the faster per-transition computational time. In future work we hope to show that fast-mixing Markov chains on the parameter space will continue to mix fast in the Firefly auxiliary variable representation.

Firefly Monte Carlo is closely related to recent ideas in using pseudo-marginal MCMC (Andrieu and Roberts, 2009) for sampling from challenging target distributions. If we sampled each of the variables $\{z_n\}$ as a Bernoulli random variable with success probability 0.5, then the joint posterior we have been using becomes an unbiased estimator of the original posterior over θ , up to normalization. Running pseudo-marginal MCMC using this unbiased estimator would be a special case of FlyMC: namely FlyMC with z and θ updated simultaneously with Metropolis-Hastings updates.

Acknowledgements

We thank Andrew Miller, Jasper Snoek, Michael Gelbart, Brenton Partridge, Oren Rippel and Elaine Angelino for helpful discussions. We also thank Alan Aspuru-Guzik, Johannes Hachmann, and Roberto Olivares-Amaya for the use of the OPV data set and introduction to the cheminformatic feature set. Partial funding was provided by Analog Devices (Lyric Labs), DARPA Young Faculty Award N66001-12-1-4219 and Samsung Advanced Institute of Technology.

References

Carlos Amador-Bedolla, Roberto R. Olivares-Amaya, Johannes Hachmann, and Alan Aspuru-Guzik. Organic photovoltaics. In K. Rajan, editor, *Informatics for Materials Science and Engineering*, pages 423–440. Elsevier, Amsterdam, 2013.

Christophe Andrieu and Gareth O Roberts. The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, pages 697–725, 2009.

Elaine Angelino, Eddie Kohler, Amos Waterland, Margo Seltzer, and Ryan P. Adams. Accelerating mcmc via

parallel predictive prefetching. In *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence*, 2014.

Rémi Bardenet, Arnaud Doucet, and Chris Holmes. Towards scaling up Markov chain Monte Carlo : an adaptive subsampling approach. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.

Dankmar Böhning. Multinomial logistic regression algorithm. *Annals of the Institute of Statistical Mathematics*, 44(1):197–200, 1992.

Johannes Hachmann, Roberto Olivares-Amaya, Sule Atahan-Evrenk, Carlos Amador-Bedolla, Roel S. Sanchez-Carrera, Aryeh Gold-Parker, Leslie Vogt, Anna M. Brockway, and Alan Aspuru-Guzik. The harvard clean energy project: Large-scale computational screening and design of organic photovoltaics on the world community grid. *The Journal of Physical Chemistry Letters*, 2(17):2241–2251, 2011.

Johannes Hachmann, Roberto Olivares-Amaya, Adrian Jinich, Anthony L. Appleton, Martin A. Blood-Forsythe, Laszlo R. Seress, Carolina Roman-Salgado, Kai Trepte, Sule Atahan-Evrenk, Suleyman Er, Supriya Shrestha, Rajib Mondal, Anatoliy Sokolov, Zhenan Bao, and Alan Aspuru-Guzik. Lead candidates for high-performance organic photovoltaics from high-throughput quantum chemistry - the harvard clean energy project. *Energy Environ. Sci.*, 7:698–704, 2014.

Matthew D. Hoffman, David M. Blei, and Francis Bach. Online learning for latent Dirichlet allocation. In *Advances in Neural Information Processing Systems 23*, 2010.

Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

Tommi S. Jaakkola and Michael I. Jordan. A variational approach to Bayesian logistic regression models and their extensions. In *Workshop on Artificial Intelligence and Statistics*, 1997.

Anoop Korattikara, Yutian Chen, and Max Welling.usterity in MCMC Land: Cutting the Metropolis-Hastings Budget. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, Department of Computer Science, University of Toronto, 2009.

Kevin P. Murphy. *Machine learning: a probabilistic perspective*. MIT Press, Cambridge, MA, 2012.

Radford M. Neal. Slice sampling. *The Annals of Statistics*, 31(3):705–767, 06 2003.

Roberto Olivares-Amaya, Carlos Amador-Bedolla, Johannes Hachmann, Sule Atahan-Evrenk, Roel S.

- Sanchez-Carrera, Leslie Vogt, and Alan Aspuru-Guzik. Accelerated computational discovery of high-performance materials for organic photovoltaics by means of cheminformatics. *Energy Environ. Sci.*, 4: 4849–4861, 2011.
- Martyn Plummer, Nicky Best, Kate Cowles, and Karen Vines. CODA: Convergence diagnosis and output analysis for MCMC. *R News*, 6(1):7–11, 2006. URL <http://CRAN.R-project.org/doc/Rnews/>.
- Rajesh Ranganath, Sean Gerrish, and David M. Blei. Black box variational inference. In *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics*, 2014.
- Gareth O Roberts and Jeffrey S Rosenthal. Optimal scaling of discrete approximations to Langevin diffusions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 60(1):255–268, 1998.
- Gareth O Roberts and Richard L Tweedie. Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli*, pages 341–363, 1996.
- Gareth O. Roberts, Andrew Gelman, and Walter R. Gilks. Weak convergence and optimal scaling of random walk Metropolis algorithms. *Annals of Applied Probability*, 7: 110–120, 1997.
- Chong Wang and David M Blei. Variational inference in nonconjugate models. *The Journal of Machine Learning Research*, 14(1):1005–1031, 2013.
- Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning*, 2011.

Sequential Bayesian Optimisation for Spatial-Temporal Monitoring

Roman Marchant
NICTA & University of Sydney
Sydney, Australia
roman.marchant@sydney.edu.au

Fabio Ramos
NICTA & University of Sydney
Sydney, Australia
fabio.ramos@sydney.edu.au

Scott Sanner
NICTA & ANU
Canberra, Australia
ssanner@nicta.com.au

Abstract

Bayesian Optimisation has received considerable attention in recent years as a general methodology to find the maximum of costly-to-evaluate objective functions. Most existing BO work focuses on *where* to gather a set of samples without giving special consideration to the sampling sequence, or the costs or constraints associated with that sequence. However, in real-world sequential decision problems such as robotics, the order in which samples are gathered is paramount, especially when the robot needs to optimise a temporally non-stationary objective function. Additionally, the state of the environment and sensing platform determine the type and cost of samples that can be gathered. To address these issues, we formulate *Sequential Bayesian Optimisation* (SBO) with side-state information within a *Partially Observed Markov Decision Process* (POMDP) framework that can accommodate discrete and continuous observation spaces. We build on previous work using *Monte-Carlo Tree Search* (MCTS) and *Upper Confidence bound for Trees* (UCT) for POMDPs and extend it to work with continuous state and observation spaces. Through a series of experiments on monitoring a spatial-temporal process with a mobile robot, we show that our UCT-based SBO POMDP optimisation outperforms myopic and non-myopic alternatives.

1 INTRODUCTION

Bayesian Optimisation (BO) [1, 6, 10] is a global optimisation technique that has recently gained popularity in the machine learning community. BO possesses major advantages when used to find the maximum of partially observed objective functions that are costly to evaluate, lack gradient information, and can only be inferred indirectly from noisy

observations. BO is robust to this setting because it builds a statistical model over the objective. More specifically, it places a *prior* over the space of functions and combines it with noisy samples to produce an incremental prediction for the unknown function. The prior usually takes the form of a *Gaussian Process* (GP) [15], which has proved successful in modelling spatial-temporal data [4, 9, 17]. The key component for the effectiveness of BO is the use of an *Acquisition Function* (AF) that guides the search for the optimum by selecting the locations where samples are gathered based on the posterior in each iteration.

BO can be readily applied to scenarios where the objective function does not vary in time and sampling locations can be chosen freely within the input domain. In real-world robotics applications, functions are likely to change with time [11] indicating that *when* to sample is as important as *where* to sample. Another important aspect in realistic settings is that the state of the environment and sampling platform determines the reachable space for gathering the next sample. Combined, these issues create an imperative for finding optimal *sequences* of sampling locations.

Most of the existing work focuses on myopic decision-making by evaluating one-step lookahead for objective sampling. Non-myopic solutions have been proposed in [5, 12], but the authors acknowledge they are considerably expensive to evaluate and do not account for possible side-state presence due to external conditions. An optimal solution to non-myopic decision-making with side-state can be formalised in the *Partially Observed Markov Decision Process* (POMDP) framework. The key here is to consider the state as a tuple, consisting of the unknown function and the state of a sensing robot. However, this leaves open the question of how one can efficiently solve the resulting POMDP.

The *online* setting for POMDP planning has received increased attention in recent years for helping overcome perceived efficiency limitations of POMDP solutions [16]. Silver and Veness [18] show how to use UCT for large POMDPs, however, this does not extend to continuous observations (without sampling). Porta *et al* [14] present

Point-Based Value Iteration (PBVI) for continuous state, action and observation POMDPs, however, this approach aims for a closed-form value function that generalises over all states, which can only be computed in more restricted cases than the general sequential BO POMDP framework we would like to propose in this work. A first connection between BO and POMDPs has been noted by [20], that solved the two-step lookahead without any efficient strategies, or considering the side-state as we do. In this work we intend to build on both [18] and [20] to apply UCT to a general POMDP formulation of SBO with side-state and continuous observations.

We begin by briefly describing *Gaussian Processes* (GPs), BO and POMDPs. We show the connections between SBO and POMDPs, followed by possible online solutions for multi-step lookahead in POMDPs that aim to provide an optimal sequence of sampling locations. In section 4 we evaluate our model for spatial-temporal monitoring problems that clearly demonstrate the benefits of our UCT algorithm for non-myopic SBO optimisation.

2 BACKGROUND

We start with a brief description of Gaussian processes as the underlying regression technique for Bayesian optimisation. We then describe BO and define notation for our POMDP formulation.

2.1 GAUSSIAN PROCESSES

A GP is a collection of random variables with a joint Gaussian distribution. A GP places a Gaussian prior over the space of functions and is completely defined by a mean function, $m(\mathbf{x})$, and a positive semi-definite covariance function $k(\mathbf{x}, \mathbf{x}')$, where \mathbf{x} is an input in a D dimensional space, $\mathbf{x} \in \mathcal{R}^D$. A latent noisy function f can be represented as $f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$. Further, we assume an additive noise model $y = f(\mathbf{x}_i) + \epsilon$ for noisy observations y from f , where $\epsilon \stackrel{iid}{\sim} \mathcal{N}(0, \sigma_n^2)$ is an independent Gaussian noise.

Given a set of N training inputs $X = \{\mathbf{x}_i\}_{i=1}^N$ and corresponding outputs $\mathbf{y} = \{y_i\}_{i=1}^N$ we can calculate the predictive distribution of f at an unknown query location \mathbf{x}^* by computing the posterior $p(f(\mathbf{x}^*)|\mathbf{y}, X, \mathbf{x}^*)$. For a GP, this predictive distribution is Gaussian, $f(\mathbf{x}^*) \sim \mathcal{N}(\bar{f}^*, \text{cov}(f^*))$, where

$$\begin{aligned} \bar{f}^* &= K(\mathbf{x}^*, X)K_X^{-1}(\mathbf{y} - m(X)), \\ \text{cov}(f^*) &= K(\mathbf{x}^*, \mathbf{x}^*) - K(\mathbf{x}^*, X)K_X^{-1}K(X, \mathbf{x}^*), \end{aligned} \quad (1)$$

and $K(A, B)$ is a covariance matrix whose element (i, j) is calculated as $k_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$, with $\mathbf{x}_i \in A$ and $\mathbf{x}_j \in B$. $K_X = K(X, X) + \sigma_n^2 I$ is the covariance matrix between observations, with identity matrix I .

The parameters of the mean and covariance functions can be estimated automatically by maximising the marginal likelihood of the data [15]. Since we will be dealing with space-time inputs, \mathbf{x} can be represented by space and time components, covariance functions can be separable [19] and learn periodic patterns [15, 21].

2.2 BAYESIAN OPTIMISATION

BO is an optimisation technique for finding the optimum $\hat{\mathbf{x}} \in \mathcal{R}^D$ of an unknown, costly to evaluate and noisy function $f : \mathcal{R}^D \rightarrow \mathbb{R}$,

$$\hat{\mathbf{x}} = \arg \max_{\mathbf{x}} f(\mathbf{x}). \quad (2)$$

In this setting f is not directly observable but we have available noisy samples from f , i.e. the i th observation can be seen as $y_i = f(\mathbf{x}_i) + \epsilon$, where $\epsilon \stackrel{iid}{\sim} \mathcal{N}(0, \sigma_n^2)$ is the noise associated to each independent observation. BO uses a GP to model f which is incrementally updated at every iteration, as new observations become available. The benefit of this approach is that for every optimum candidate location we can evaluate an analytical expression for the expected value of f and its variance (Equation 1). This information is used by an *Acquisition Function* (AF), $h(\mathbf{x})$, whose purpose is to guide the search for the optimum. At each iteration, one sample is gathered from f at a location selected by maximising $h(\mathbf{x})$, which is a simpler and faster optimisation procedure (compared to the original problem of optimising f). Algorithm 1 presents the BO algorithm.

Algorithm 1 Bayesian Optimisation

Inputs: f, h

Outputs: $\hat{\mathbf{x}}, f(\hat{\mathbf{x}})$

```

for  $j = 1, 2, 3, \dots$  {Max iterations} do
  Find  $\mathbf{x}_j = \arg \max_{\mathbf{x}} h(\mathbf{x})$ 
   $y_j \leftarrow f(\mathbf{x}_j)$  ▷ Gather sample from  $f$ 
  Augment training set with  $(\mathbf{x}_j, y_j)$ .
  Update GP
  if  $y_j > \mu(\hat{\mathbf{x}})$  then
     $\hat{\mathbf{x}} \leftarrow \mathbf{x}_j$  ▷ Update location of optimum
  end if
end for
```

Many AFs have been proposed on the literature [6–8, 12]. In this paper we use the *Upper Confidence Bound* (UCB) [3] acquisition function, however, none of the algorithms presented here are strongly linked to this specific AF.

2.3 PARTIALLY OBSERVABLE MDPs

POMDPs are a unified framework for sequential decision making under uncertainty when the state is not directly ob-

servable. A POMDP is fully represented by the following tuple $\langle S, A, T, R, Z, O \rangle$, where:

- S : Set of states $\{s_1, s_2, \dots, s_n\}$.
- A : Set of actions $\{a_1, a_2, \dots, a_n\}$.
- $T : S \times A \times S \rightarrow [0, 1]$ is a transition function that represents the probability of transition between states s and s' when executing action a , i.e. $T(s, a, s') = p(s'|s, a)$.
- $R : S \times A \rightarrow \mathbb{R}$ is a reward function that encodes the reward of executing action a on state s , i.e. $R(s, a)$.
- Z : Finite set of observations $\{z_1, z_2, \dots, z_n\}$.
- $O : S \times A \times Z \rightarrow [0, 1]$ is a observation function that represents the probability of observing o if action a is executed with resulting state s , i.e. $O(o, a, s) = p(o|a, s)$.

In POMDPs, it is well-known that a belief state summarises *all* relevant information in the observation history of a POMDP. Given a belief state $b_{t-1}(s)$, the belief at time t can be updated as:

$$b_t(s') \propto P(o|s') \int b_{t-1}(s) P(s'|s, a) ds. \quad (3)$$

where $b_0(s) = p(s)$ represents the initial belief state.

Solving a POMDP is equivalent to determining a policy π^* mapping belief states to actions which maximizes some objective criterion. An optimal policy over an infinite horizon can be found by maximising the expected cumulative discounted reward r_t (for discount $\gamma \in (0, 1]$) at time step t when executing π starting from belief state $b_0 := b_0(s)$,

$$\pi^* = \arg \max_{\pi} E \left[\sum_{t=0}^{\infty} \gamma^t \cdot r_t^{\pi} | b_0 \right]. \quad (4)$$

3 SEQUENTIAL BAYESIAN OPTIMISATION

With the definitions above we can now extend BO to a sequential setting. In order to apply BO to more realistic problems we expand the existing theory to a more generic framework and include the notion of state in the definition of the problem. This means that at every step a generic reward, r , can be obtained by sampling at \mathbf{x} . This reward depends on the state \mathbf{x} of a mobile robotic sensor and the expected value of the objective function $f(\mathbf{x})$. In the general case, because gathering each sample has an associated reward, the order in which they are gathered has a direct influence over the total accumulated reward for a specific lookahead. We call this kind of optimisation technique *Sequential Bayesian Optimisation* (SBO).

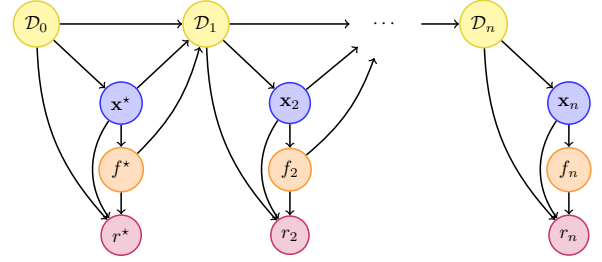


Figure 1: Bayesian network representation for SBO.

Sampling locations and their associated observations are grouped in \mathcal{D} , which is built incrementally as shown in Figure 1. Using a similar treatment to plain BO, the myopic expectation of the reward r (ER), can be obtained by marginalising out all unknown outcomes,

$$\text{ER}(\mathbf{x}^* | \mathcal{D}_0) = \mathbb{E}_{f^*} [r(\mathbf{x}^*, f^* | \mathcal{D}_0)] \quad (5)$$

$$= \int r(\mathbf{x}^*, f^* | \mathcal{D}_0) p(f^* | \mathbf{x}^*, \mathcal{D}_0) df^*. \quad (6)$$

The n -step lookahead expression is given by

$$\begin{aligned} \text{ER}_n(\mathbf{x}^* | \mathcal{D}_0) &= \int \dots \int \left(r(\mathbf{x}^*, f^* | \mathcal{D}_0) + \sum_{i=2}^n (r(\mathbf{x}_i, f_i | \mathcal{D}_{i-1})) \right) \\ &\quad p(f^* | \mathbf{x}^*, \mathcal{D}_0) \times \prod_{i=2}^n p(f_i | \mathbf{x}_i) p(\mathbf{x}_i | \mathcal{D}_{i-1}) \\ &\quad df^* df_2 \dots df_n d\mathbf{x}_2 \dots d\mathbf{x}_n, \end{aligned} \quad (7)$$

where we are marginalising out all future outcomes ($f^*, f_2 \dots f_n$) and locations ($\mathbf{x}_2 \dots \mathbf{x}_n$). This expression has been derived in [12], however, it presents a slight modification because we are considering the whole sequence of locations for reward calculation, not just the expected improvement for the last sample. It is important to note that within the BO algorithm, ER can be seen as the acquisition function $h(\mathbf{x})$ for selecting sampling locations. ER needs to be maximised w.r.t. \mathbf{x}^* in each iteration of the algorithm.

In real robotic deployments, decisions $\{\mathbf{x}_i\}$ can be represented as continuous paths followed by the robot. We represent these paths as parametrised curves, \mathcal{C} , over the input space, with each curve characterised by a set of parameters Θ . The following expression shows the expected reward for traversing a path with parameters Θ^* , and looking ahead for n steps, i.e. considering n paths in the future and

integrating all possible rewards,

$$\begin{aligned}
& \text{ER}_n(\Theta^*, \mathcal{D}_0) \\
&= \int_{f^*} \int_{f_2} \cdots \int_{f_n} \int_{\Theta_2} \cdots \int_{\Theta_n} \\
&\quad \left(r(\mathcal{C}_{\Theta^*} | \mathcal{D}_{N-1}) + \sum_{i=2}^n r(\mathcal{C}_{\Theta_i} | \mathcal{D}_{i-1}) \right) \\
&\quad p(f^* | \Theta^*, \mathcal{D}_{N-1}) \prod_{i=2}^n p(f_i | \Theta_i, \mathcal{D}_{i-1}) p(\Theta_i | \mathcal{D}_{i-1}) \\
&\quad df^* df_2 \cdots df_n d\Theta_2 \cdots d\Theta_n
\end{aligned} \tag{8}$$

In this expression we are marginalising out all possible observations and paths for n steps. Unfortunately, given the infinite number of possible paths, this integral does not have an analytical solution and can only be approximated. In the following section we illustrate how SBO can be represented in a POMDP formulation and solved using online decision making POMDP solvers.

3.1 SBO AS ONLINE POMDPs

Our SBO formulation is *state-aware*, i.e. it considers the state of a mobile robot for decision making. This problem can be formulated as a POMDP problem in a similar manner as described in [20] for regular BO. The main idea is to include the objective function, which is partially observable, together with the state of the robot, into the state definition. We assume the robot's pose is fully observable and part of the state as side information \mathbf{p} . The decision of where to sample f is encoded by the action space, which is limited by the possible actions that can be performed by the robot. In the discrete case, an action is represented by moving to a specific cell. For the continuous case an action means travelling along a continuous path. More formally, the elements of the POMDP definition for side-state SBO are:

- S : The state which is a tuple $\{f, \mathbf{p}\}$, where f is a latent (not directly observable) function defined over space and time representing the unknown process. Additionally, we include the state of the sensing robot, \mathbf{p} , which is fully observable, as the side information.
- A : The parametrised action space $a(\Theta)$. The actions can be described as move according to parameters Θ and gather a samples from f in the process. For the discrete sampling case, Θ represents a location in the spatial domain of f . For the continuous case, Θ are the parameters of a continuous curve defined over the domain of f .
- T : The transition function which is defined over the entire state $\{f, \mathbf{p}\}$. $T(\{f, \mathbf{p}\}, a(\Theta), \{f', \mathbf{p}'\})$ is the transition probability of resulting in state $\{f', \mathbf{p}'\}$ given that action $a(\Theta)$ was taken at state $\{f, \mathbf{p}\}$. Assuming that the robot does not affect or change the

objective function, the joint transition probability can be decomposed into the product of two independent transition functions:

$$\begin{aligned}
& T(\{f, \mathbf{p}\}, a(\Theta), \{f', \mathbf{p}'\}) \\
&= T_f(f, a(\Theta), f') T_{\mathbf{p}}(\mathbf{p}, a(\Theta), \mathbf{p}')
\end{aligned} \tag{9}$$

Since f is not affected by the actions in A , the transition function T_f is the identity.

$$T_f(f, a(\Theta), f') = \delta(f' - f). \tag{10}$$

The transition function $T_{\mathbf{p}}$ depends on the definition of the action space, and can often be modeled deterministically since robots can navigate with accurate positioning and path following controllers in many large-scale outdoor monitoring applications. When the action space is defined as a location, the action parameters Θ represent a location, and $T_{\mathbf{p}}$ can be calculated using

$$T_{\mathbf{p}}(\mathbf{p}, a(\Theta), \mathbf{p}') = \delta(\mathbf{p}' - \Theta) \tag{11}$$

- R : If the objective function f is sampled at Θ then the *expected* reward in an SBO POMDP belief state is the objective value w.r.t. beliefs $b(f)$ minus any application-specific action *cost*(\mathbf{p}, Θ) associated with moving from \mathbf{p} to Θ :

$$\text{ER}(\{f, \mathbf{p}\}, a(\Theta)) = \mathbb{E}_{b(f)}[f(\mathbf{x})] + \text{cost}(\mathbf{p}, a(\Theta)) \tag{12}$$

When the action space is parametrised as locations the reward can be evaluated directly. However, when the action space is parametrised by curves, the reward associated to an action is given by the sum of the rewards along the curve \mathcal{C} :

$$R(\{f, \mathbf{p}\}, \mathcal{C}(\Theta)) = \sum_{\mathbf{x} \in \mathcal{C}(\Theta)} R(\{f, \mathbf{p}\}, \mathcal{C}(\Theta)), \tag{13}$$

where the sum can be replaced by an integral when the sensing device allows continuous sampling along the curve.

- Z : In SBO, objective observations $z \in \mathbb{R}$ are simply noisy observations of $f(\Theta)$ as defined next.
- O : The observation function is defined according to the action space parametrisation. When the action space is defined as a sampling location Θ , f can be evaluated directly on Θ .

$$O(z, a(\Theta), \{f, \mathbf{p}\}) = p(z | f(\mathbf{x} = \Theta)) \tag{14}$$

We observe that for GPs, we can generate z by sampling from a GP marginal for f at location Θ . When the action space is a curve \mathcal{C} , f is evaluated at a number of sample locations within \mathcal{C} . The observation function for this set of observations $\{z_i\}$ is

$$O(\{z_i\}, \mathcal{C}(\Theta), \{f, \mathbf{p}\}) = \prod_{\mathbf{x}_i \in \mathcal{C}(\Theta)} p(z_i | f(\mathbf{x}_i)) \tag{15}$$

The belief is then the probability distribution over the space of functions f and updated as described in equation (3). If the model for f is a GP, the belief update for an action-observation pair can be computed directly. The action component can be ignored for purposes of updating a belief in f , since as stated earlier, the robot’s physical state does not affect or change the objective function, it only restricts the observations that can be made regarding f . Therefore, the belief update over f is simply computed by adding new location-observation pairs to the GP training data set.

Next, we present a methodology to solve this POMDP by sampling a subset of action primitives that the robot can execute in the environment. Action primitives and maximum likelihood observation selection are the key points to approximate Equation 8.

3.2 MCTS AND UCT FOR SBO

MCTS is a popular technique for solving large POMDPs [2, 18]. This method can turn a tedious search in decision trees into an efficient approximation using Monte-Carlo samples from the tree. MCTS efficiently searches reachable beliefs from a given initial belief state and is useful for real-time online planning. Its main advantage over other techniques, such as Point-Based Value Iteration is that it does not require the overhead of maintaining alpha-functions over all states nor choosing the states for which alpha-functions should be maintained.

[18] have shown how MCTS can reach impressive scalability through the use of UCT, which they call POMCP. In this work we conserve their idea of efficient tree search. However, we consider the case where the belief update is a GP update for f and use the maximum likelihood observation, as it is done by [13]. The maximum-likelihood observation assumption helps reducing the branching factor of the tree, which would grow uncontrollably when sampling observations.

For the SBO problem, each node in the tree consists of a belief representation for f and a side-state \mathbf{p} . We define the i th node by v_i . For each action-observation pair, the belief representation $b(f)$ and side state \mathbf{p} are updated easily since $b(f)$ is a GP prior and side-state transitions are deterministic and observable. Every new action-observation simulation creates a new node with the updated belief and side-state.

The tree is built incrementally starting with an initial node v_0 . Figure 2 shows an example of a small tree that has been expanded partially with two action primitives. Each ellipse represents a node, that consists of a belief over f , $b(f)$, and side-state \mathbf{p} . A node is expanded by simulating the outcomes of executing an action. The outcomes (noisy observations of f) are the maximum-likelihood observations. The branching factor of the tree will be the number of ac-

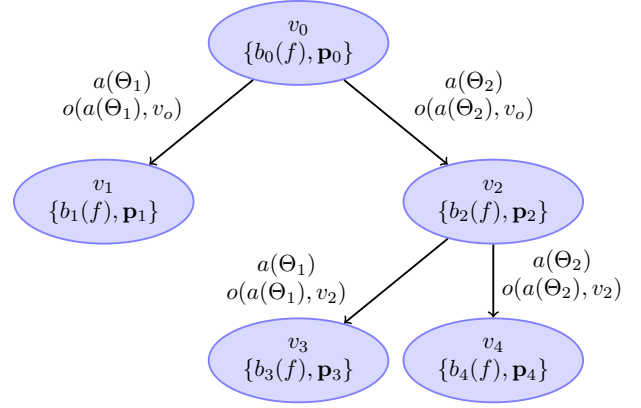


Figure 2: Example of a tree with depth 2, partially expanded from a set of two action primitives.

tion primitives. When a node is expanded, a new node is created using the updated belief and new side-state.

The first step in each iteration is to find a leaf node candidate for expansion/evaluation, which is done inside of the function TREEPOLICY. This search is guided by the function BESTCHILD, which uses the statistics stored for each node (accumulated reward and number of visitations) to select the most *promising* child. Starting from the chosen leaf node, a random action selection is conducted until the maximum depth is reached, executed within DEFAULTPOLICY. The total accumulated reward is then backed up in function BACKUP, that updates the statistics on all the nodes visited during the current iteration. Each iteration of the search algorithm simulates a sequence of up to n actions, where n is the maximum depth. When the iteration loop is completed, the best action is determined by picking the best child from the parent node v_0 . Algorithm 2 shows the full procedure for building a tree and returning the best immediate action.

4 EXPERIMENTS

In this section we present experiments where a robot attempts to learn the behaviour of a spatial-temporal process by choosing actions that maximise the expected reward. We show comparisons for two different problems, including one with time dependent behaviour.

For illustrative purposes we simulate 2D functions in space that can change with time, such that,

$$f : \mathbb{R}^3 \rightarrow \mathbb{R} \\ (x_1, x_2, t) \rightarrow y.$$

In these experiments, the pose $\mathbf{p} = (x_{1r}, x_{2r}, \theta_r)$ of a robot is the side-state for the SBO formulation and f is the unknown function to be estimated. The belief $b(f)$ is represented by a GP using a separable space-time covariance function [19]. The structure of the GP’s covariance function can capture periodicity in f from the training data.

Algorithm 2 Monte Carlo Tree Search for SBO

```

function  $a^* = \text{MCTS}(b(f), \mathbf{p}, \text{depth}_{\max})$ 
   $v_0 = \text{NEWNODE}(b(f), \mathbf{p}, \text{reward}_{\min})$ 
   $i \leftarrow 0$ 
  while  $i < \{\text{Max MCTS iterations}\}$  do
     $v_l \leftarrow \text{TREEPOLICY}(v_0)$ 
     $r \leftarrow \text{DEFAULTPOLICY}(v_l)$ 
     $\text{BACKUP}(v_l, r)$ 
  end while
  return  $a^* = \text{BESTCHILD}(v_0)$ 
end function
function  $v_l = \text{TREEPOLICY}(a)$ 
   $v \leftarrow v_0$ 
  while  $\text{DEPTH}(v) \leq \text{depth}_{\max}$  do
    if  $v$  has untried actions then
      Choose  $a$  from untried actions
       $r \leftarrow \text{Simulate } a$   $\triangleright$  Simulate Reward
      Update  $b(f)$  and  $\mathbf{p}$ .
      return  $v_l = \text{NEWNODE}(b(f), \mathbf{p}, r)$ 
    else
       $v = \text{BESTCHILD}(v)$ 
    end if
  end while
  return  $v$ 
end function
function  $r = \text{DEFAULTPOLICY}(v_l)$ 
   $r \leftarrow$  Get reward accumulated until  $v_l$ 
   $d \leftarrow \text{DEPTH}(v_l)$ 
  while  $d \leq \text{depth}_{\max}$  do
    Select  $a$  randomly
    Update  $b(f)$  and  $\mathbf{p}$ .
     $r_a \leftarrow \text{Simulate } a$ 
     $r \leftarrow r + r_a$ 
     $d \leftarrow d + 1$ 
  end while
end function
function  $\text{BACKUP}(v_l, r)$ 
   $v \leftarrow v_l$ 
  while  $v \neq v_0$  do
    Increase visited counter for  $v$ 
    Increase accumulated reward for  $v$ 
     $v \leftarrow \text{PARENT}(v)$ 
  end while
end function
function  $v_c = \text{BESTCHILD}(v_p)$ 
   $V \leftarrow$  Children of  $v_p$ 
  for  $v_i \in V$  do
     $N_p \leftarrow$  Visited counter of  $v_p$ 
     $N_i \leftarrow$  Visited counter of  $v_i$ 
     $R_i \leftarrow$  Accumulated reward
     $g(i) = \frac{R_i}{N_i} + \kappa_{MC} \sqrt{\frac{2\ln(N_p)}{N_i}}$ 
  end for
   $v_c \leftarrow \arg \max_{v_i \in V} g(i)$ 
end function

```

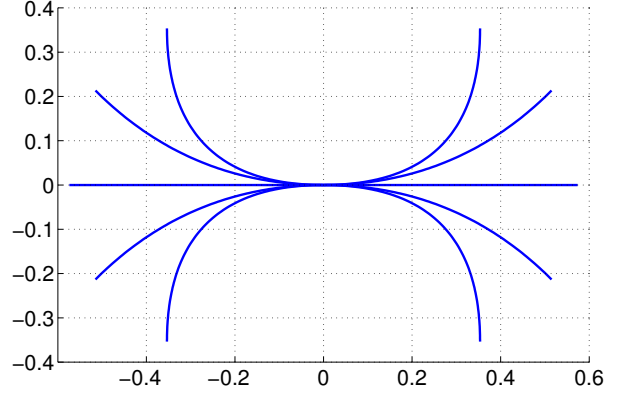


Figure 3: Motion primitives for a mobile robot.
Axis in km.

Since the robot travels at a certain speed $\dot{\mathbf{p}}$, the reachable area for sampling f depends on the side-state \mathbf{p} .

The action space A is determined by a set of motion primitives parametrised as 2D cubic splines. A cubic spline \mathcal{C} is a continuous function mapping from \mathbb{R} to \mathbb{R}^2 , $\mathcal{C}(u|\Theta) = [\mathcal{C}_{x_1} \ \mathcal{C}_{x_2}]^T$, with $u \in [0, 1]$, defined as

$$\mathcal{C} = \Theta \begin{bmatrix} u^3 & u^2 & u^1 & 1 \end{bmatrix}^T, \quad (16)$$

where Θ are the parameters expressed as a 2×4 matrix for the 2D case. With appropriate parametrisation, the curves generate the ten primitives $A = \{\mathcal{C}_i\}_{i=1 \dots 10}$ shown in Figure 3 for $\mathbf{p} = \mathbf{p}_0 = (0, 0, 0)$. For values of $\mathbf{p} = (x_{1r}, x_{2r}, \theta_r)$ the curves are rotated and translated using translation and rotation matrices. We define a transition function $T_{\mathbf{p}}(\mathbf{p}, \mathcal{C}_i, \mathbf{p}') = 1$ for a cubic spline transformed from \mathbf{p} , with

$$\mathbf{p}' = \left(\mathcal{C}_i(u=1)_{x_1}, \mathcal{C}_i(u=1)_{x_2}, \frac{\partial \mathcal{C}_{x_1} / \partial u}{\partial \mathcal{C}_{x_2} / \partial u} \bigg|_{u=1} \right). \quad (17)$$

Before an action (curve) is selected for execution, the robot computes the optimal action using the MCTS algorithm (Algorithm 2). The robot gathers noisy samples from f along \mathcal{C} while the action is being executed.

4.1 STATIC FUNCTION

In the first example, we simulate a static function, with expression

$$y = f(x_1, x_2, t) = e^{-(x_1-4)^2} e^{-(x_2-1)^2} + 0.8e^{-(x_1-1)^2} e^{-\left(\frac{x_2-4}{2.5}\right)^2} + 4e^{-\left(\frac{x_1-10}{5}\right)^2} e^{-\left(\frac{x_2-10}{5}\right)^2}, \quad (18)$$

where $x_1 \in [0, 5]$, $x_2 \in [0, 5]$, and $t \in [0, \infty]$. Figure 4 shows a plot for this function, where it is easy to distinguish two main peaks with different amplitudes. The robot is initially located at pose $\mathbf{p} = (0.5, 0.5, 0)$ and travels at a fixed speed of $0.2m/s$, gathering a sample every 5 minutes.

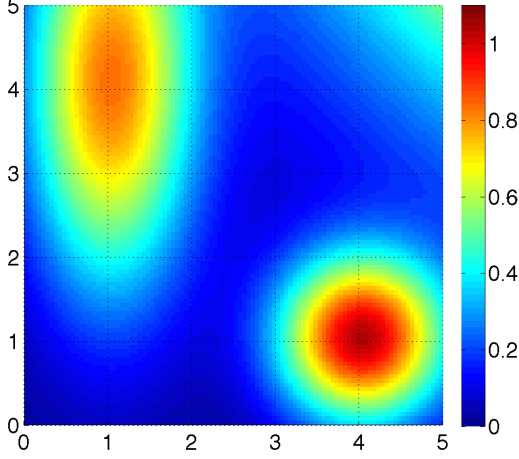


Figure 4: Static goal function. Axis in km.

We first want to evaluate how the definition of the reward function R within the POMDP context impacts the action selection properties of the algorithm. We compare four different reward functions based on the UCB acquisition function, $r(\mathbf{x}) = \mu(\mathbf{x}) + \kappa\sigma(\mathbf{x})$, where $\kappa_i \in \{1.0 \times 10^6, 200, 20, 10\}$. It is a well known fact that the value of κ affects the exploration-exploitation trade off and this is clearly reflected in the resulting paths followed by each robot, as shown in Figure 5. The most explorative path sequence corresponds to $\kappa = 1.0 \times 10^6$ (Figure 5a) and the least explorative is $\kappa = 10$ (Figure 5d). Between these two extremes there are intermediate solutions where exploitation is favoured more strongly for lower values of κ .

In the next experiment we focus on the depth of the action selection search, i.e. the number n of lookahead steps for decision making. This corresponds to the maximum depth allowed in the search tree. We first evaluated the entire decision tree, which means simulating all the possible combinations of actions. This approach, which we call *Full Tree* (FT) strategy, will need $|A|^n$ simulations which becomes impractical quickly. In fact, for this paper we only consider FT strategies with $n \leq 3$. We compare the performance of FT against MCTS (Algorithm 2) where the number of simulations is a parameter. Clearly, for a same depth, MCTS is bounded by FT, however MCTS can find near-optimal solutions much faster. For this reason we were able to experiment with depths up to $n \leq 5$. We compare six different combinations of depth and algorithm type as indicated in Table 1.

The reward function used for these simulations was $r(\mathbf{x}) = \mu(\mathbf{x}) + 1.0 \times 10^6 \sigma(\mathbf{x})$ for all cases. Therefore, the only difference in action selection is due to the number of lookahead steps. Figure 6 shows the paths followed by the robot at $t = 2.3$ days, when it had already gathered 616 samples from f . This figure does not show all cases, only the four most relevant ones. It is interesting to observe that

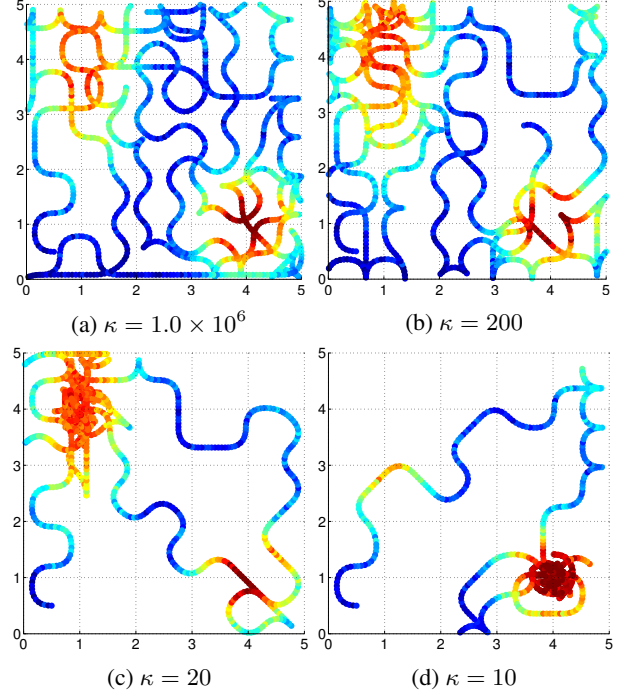


Figure 5: Comparison of followed paths for purely exploration behaviour using Full Tree and MCTS-UCT. Axis in km. Colour scale represents the value of sampled values.

Table 1: Experiment for Depth and Algorithm Type Comparison

Id	Algorithm	Max Depth	Iterations
1	FT	1	10
2	FT	2	110
3	FT	3	1110
4	MCTS	3	100
5	MCTS	4	150
6	MCTS	5	400

the search with FT Depth 1 (Figure 6a) has not achieved a full coverage of the area and is highly susceptible to get trapped and collide into the edges of the domain, which is clearly sub-optimal from an exploration point of view. On the other hand, the FT Depth 2 shows increased coverage capability, which is improved further for deeper search strategies. FT Depth 3 and MCTS Depth 3 show similar result, with the clear advantage that MCTS requires only 10% of the number of simulations of FT.

We also compare the accumulated reward over time for each case in Figure 7. This illustrates the advantage of using a multi-step lookahead strategy in increasing the total accumulated reward. However, it is not clear the advantage of using higher depths than two, as they do not show a clear improvement in accumulated reward. The main rea-

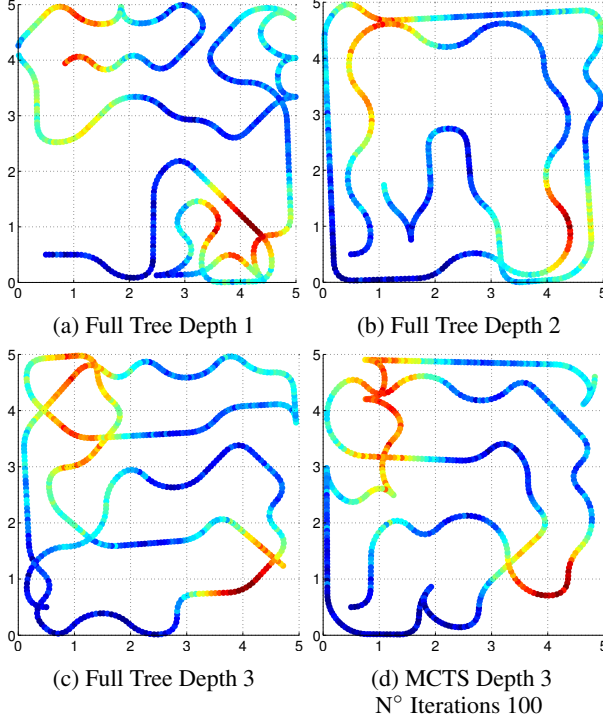


Figure 6: Comparison of paths for purely exploration behaviour using FT and MCTS-UCT. Axis in km. Colour scale represents the value of samples.

son behind this is that f does not change over time, thus making the problem simple enough such that any strategy with depth greater than 1 would be very close to the optimal solution.

4.2 DYNAMIC FUNCTION

In this second experiment we use a more complex function that changes over time,

$$y = f(x_1, x_2, t) = e^{-\left(\frac{x_1 - 2 - f_1(t)}{0.7}\right)^2} e^{-\left(\frac{x_2 - 2 - f_2(t)}{0.7}\right)^2}, \quad (19)$$

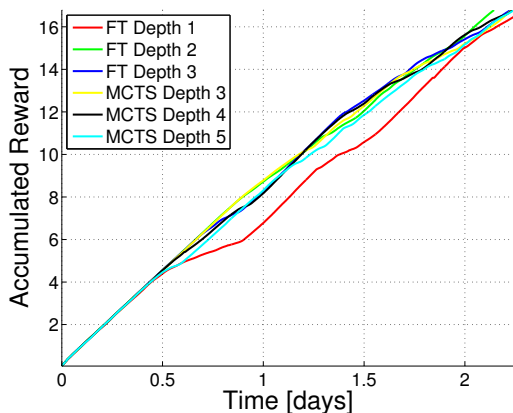


Figure 7: Accumulated reward for static goal function.

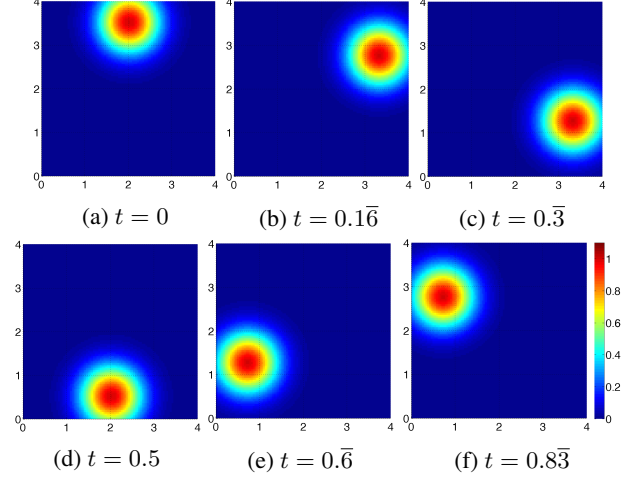


Figure 8: Dynamic goal function within one period. Axis in km.

with $f_1(t) = 1.5 \sin(2\pi t)$, $f_2(t) = 1.5 \cos(2\pi t)$, $x_1 \in [0, 5]$, $x_2 \in [0, 5]$, and $t \in [0, \infty]$. This expression generates a function where the peak moves over time. The peak circles clockwise around $(x_1, x_2) = (2, 2)$ periodically, with a period of 1 day. The motivation for this example comes from air pollution monitoring tasks where we are interested in following peaks of pollution through time while learning how the entire process evolves in space-time. Figure 8 shows the goal function for 6 time steps within one period.

Similarly to the previous experiment, the robot is initially located at pose $\mathbf{p} = (0.5, 0.5, 0)$, travels at a fixed speed of 0.12m/s and gathers a sample every 15 minutes. The goal in this experiment is to find and follow the maximum of f over time. Therefore, we select the reward function $r(\mathbf{x}) = \mu(\mathbf{x}) + 10\sigma(\mathbf{x})$, which according to Figure 5, should generate paths concentrated over the maximum values of f . Ideally, the robot should learn to follow the peak through time which would be possible for speeds greater than 0.109m/s . We try the same set of depth-algorithm pairs as in Section 4.1 and detailed in Table 1. We only show results for the extreme cases with the purpose of avoiding clutter in the figures.

Figure 9 illustrates the advantage of using multi-step lookahead strategies. The first row shows paths for FT Depth 1, where it can be seen that the robot does not learn how to follow the peak around a circle within 15 days. The second row, MCTS Depth 2, which only does 15 more simulations per iteration than FT Depth 1, the robot is already able to learn the circular pattern at $t = 12$ days. With deeper search strategies, the time required to learn the pattern decreases significantly indicating a better exploration and exploitation solution. In fact, for MCTS Depth 5 the pattern is learnt in $t = 8$ days.

Figure 10 shows the benefits of using non-myopic strategies for action selection. The cumulative reward is clearly

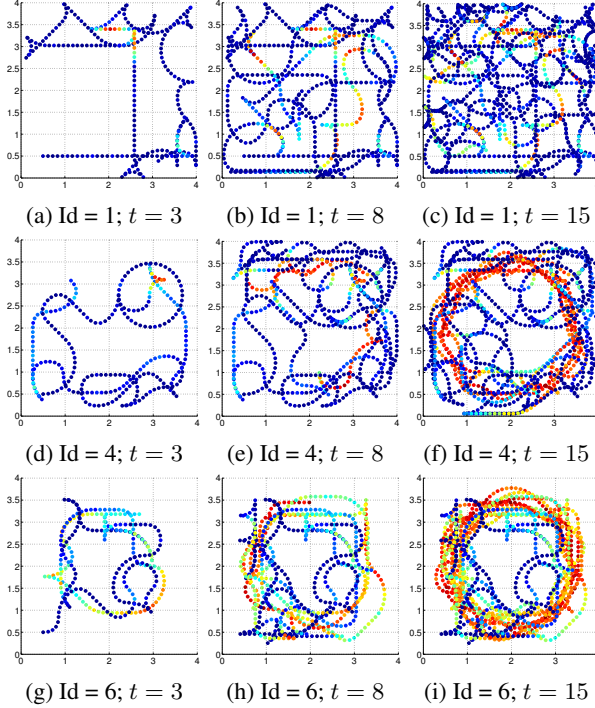


Figure 9: Comparison of followed paths for FT and MCTS-UCT in a dynamic function. First row shows the paths for FT, Depth 1; Second row shows the paths for MCTS, Depth 2; Third row shows the paths for MCTS, Depth 5. Colour scale represents the value of samples.

larger for multi-step lookahead decision making algorithms. The best solution is MCTS Depth 5, that is clearly superior for the entire duration of the simulation. A steeper slope for accumulated reward indicates that a method has learnt how to follow the peak. Then from this plot it is also clear that FT Depth 1 is not able to capture space-time dependencies properly.

It is important also to compare FT Depth 2 with MCTS Depth 2. The fact that FT is an upper bound for MCTS Depth 2 can be confirmed from Figure 10. In addition, it can be seen that both strategies accumulate similar rewards, which is a good indication that MCTS will approximate the FT solution, even with only 25% of the total tree.

Finally, Figure 11 shows how MCTS prioritises the search over promising paths. The pose of the robot at this instant is $\mathbf{p} = (1.5, 3, 0)$. Red paths are result of the function DEFAULTPOLICY that did not get further explored and blue paths are the paths present in the tree. It can be seen how the tree automatically grows towards potentially informative areas, i.e. where the reward is higher. The green curve is the best branch of the tree.

5 CONCLUSION

In this paper we proposed formulating the sequential BO problem as a POMDP. Our main contribution was to de-

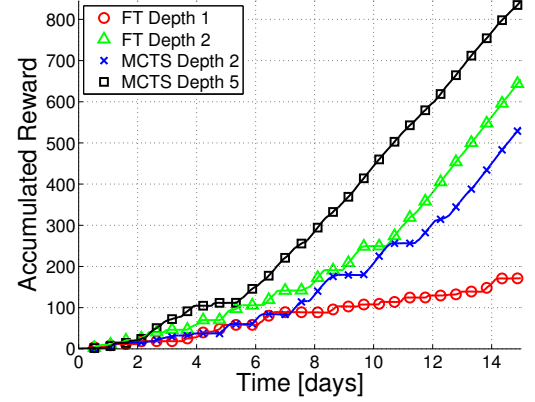


Figure 10: Accumulated reward for dynamic goal function.

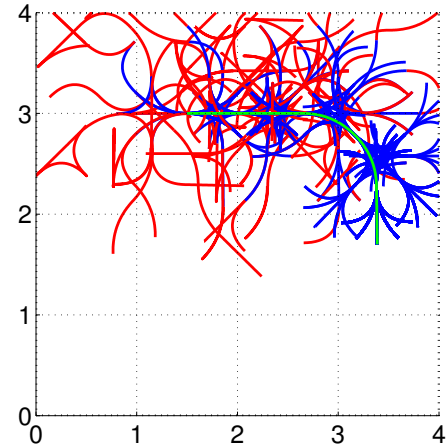


Figure 11: Example of tree built for MCTS Depth 5.

termine a non-myopic decision making solution that maximises reward and takes into account the belief of an unknown space time process and the state of a mobile robot acting as a sensor. We formulated the solution for the POMDP analogue of SBO using a modified version of the UCT algorithm for MCTS, which is a scalable and efficient way of finding asymptotically optimal decisions.

We demonstrate empirically the advantage of using non-myopic planning solutions, which becomes especially important when the objective function dynamically changes over time.

Even though long-term decision-making under uncertainty is a very complex problem, we solved it using a scalable method that works for realistic scenarios with state-dependent restrictions and time variation. We believe that using multi-step lookahead path planning is a convenient and practical way for solving many robotic problems requiring the accurate representation of real space-time phenomena, such as environment monitoring.

References

- [1] E. Brochu, V. M. Cora, and N. de Freitas. A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning. Technical Report arXiv:1012.2599, University of British Columbia, 2010.
- [2] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton. A Survey of Monte Carlo Tree Search Methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):1–43, 2012.
- [3] D. D. Cox and S. John. A Statistical Method for Global Optimization. In *IEEE Conference on Systems, Man, and Cybernetics (SMC)*, 1992.
- [4] N. Cressie and C. K. Wikle. *Statistics for Spatio-Temporal Data*. Wiley, 2011.
- [5] D. Ginsbourger, R. L. Riche, and L. Carraro. A Multi-points Criterion for Deterministic Parallel Global Optimization based on Gaussian Processes. HAL: hal-00260579, 2008.
- [6] M. Hoffman, E. Brochu, and N. de Freitas. Portfolio Allocation for Bayesian Optimization. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2011.
- [7] D. R. Jones. A Taxonomy of Global Optimization Methods Based on Response Surfaces. *Journal of Global Optimization*, pages 345–383, 2001.
- [8] D. R. Jones, M. Schonlau, and W. J. Welch. Efficient Global Optimisation of Expensive Black-Box Functions. *Journal of Global Optimization*, pages 455–492, 1998.
- [9] F. Lindgren, R. Håvard, and J. Lindström. An explicit link between Gaussian fields and Gaussian Markov random fields: the stochastic partial differential equation approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2011.
- [10] D. J. Lizotte. *Practical Bayesian Optimization*. PhD thesis, University of Alberta, 2008.
- [11] R. Marchant and F. Ramos. Bayesian Optimisation for Intelligent Environmental Monitoring. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2012.
- [12] M. A. Osborne, R. Garnett, and S. Roberts. Gaussian Processes for Global Optimization. In *International Conference on Learning and Intelligent Optimization (LION)*, 2009.
- [13] R. Platt, R. Tedrake, L. Kaelbling, and T. Lozano-Perez. Belief Space Planning Assuming Maximum Likelihood Observations. In *Robotics Science and Systems (RSS)*, 2010.
- [14] J. M. Porta, N. Vlassis, T. S. Matthijs, and P. Poupart. Point-Based Value Iteration for Continuous POMDPs. *Journal of Machine Learning Research*, 2006.
- [15] C. E. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge, Massachuset, 2006.
- [16] S. Ross, J. Pineau, S. Paquet, and B. Chaib-draa. On-line Planning Algorithms for POMDPs. *Journal of Artificial Intelligence Research*, 2008.
- [17] S. Sarkka and J. Hartikainen. Infinite-Dimensional Kalman Filtering Approach to Spatio-Temporal Gaussian Process Regression. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012.
- [18] D. Silver and J. Veness. Monte-Carlo Planning in Large POMDPs. In *Neural Information Processing Systems (NIPS)*, 2010.
- [19] A. Singh, F. Ramos, H. D. Whyte, and W. J. Kaiser. Modeling and Decision Making in Spatio-Temporal Processes for Environmental Surveillance. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2010. ISBN 978-1-4244-5038-1.
- [20] M. Toussaint. The Bayesian Search Game. *Theory and Principled Methods for the Design of Metaheuristics*, 2012.
- [21] A. G. Wilson and R. P. Adams. Gaussian Process Kernels for Pattern Discovery and Extrapolation. In *International Conference on Machine Learning (ICML)*, 2013.

AND/OR Search for Marginal MAP

Radu Marinescu
IBM Research – Ireland
radu.marinescu@ie.ibm.com

Rina Dechter and Alexander Ihler
University of California, Irvine
{dechter, ihler}@ics.uci.edu

Abstract

Marginal MAP problems are known to be very difficult tasks for graphical models and are so far solved exactly by systematic search guided by a join-tree upper bound. In this paper, we develop new AND/OR branch and bound algorithms for marginal MAP that use heuristics extracted from weighted mini-buckets enhanced with message-passing updates. We demonstrate the effectiveness of the resulting search algorithms against previous join-tree based approaches, which we also extend to accommodate high induced width models, through extensive empirical evaluations. Our results show not only orders-of-magnitude improvements over the state-of-the-art, but also the ability to solve problem instances well beyond the reach of previous approaches.

1 INTRODUCTION

Graphical models provide a powerful framework for reasoning with probabilistic and deterministic information. These models use graphs to capture conditional independencies between variables, allowing a concise representation of knowledge as well as efficient graph-based query processing algorithms. Combinatorial maximization or maximum *a posteriori* (MAP) tasks arise in many applications and often can be efficiently solved by search schemes.

The marginal MAP problem distinguishes between maximization variables (called MAP variables) and summation variables (the others). Marginal MAP is NP^{PP} -complete [1]; it is difficult not only because the search space is exponential in the number of MAP variables, but also because evaluating the probability of any full instantiation of the MAP variables is PP-complete [2]. Algorithmically, this means that the variable elimination operations (max and sum) are applied in a constrained, often more costly order.

State-of-the-art exact algorithms for marginal MAP are

typically based on depth-first branch and bound search. A key component of branch and bound search is the heuristic function; while partitioning based heuristics such as *mini-bucket elimination* (MBE) [3] or *mini-cluster-tree elimination* (MCTE) [4, 5] can be applied to the constrained elimination order, the current state-of-the-art is to use a heuristic based on an *exact* solution to an *unconstrained* ordering, introduced by Park and Darwiche [6] and then refined by Yuan and Hansen [7]. These techniques appear to work well when the unconstrained ordering results in a small *induced width*. However, in many situations this is a serious limitation. As one contribution, we extend both algorithms to use mini-bucket partitionings schemes, enabling them to be applied to a wider variety of problem instances.

Importantly however, exact algorithms for pure max- or sum-inference problems have greatly improved in recent years. AND/OR branch and bound (AOBB) algorithms explore a significantly smaller search space, exploiting problem structure far more effectively [8]. The partition-based heuristics used by AOBB have also seen significant improvements – for MAP, cost-shifting [9] can be used to tighten the heuristic, while for summation, an extension of MBE called *weighted mini-bucket* (WMB) [10] uses Hölder’s inequality and cost-shifting to significantly enhance the likelihood bounds. WMB is closely related to variational bounds such as tree-reweighted belief propagation [11] and conditional entropy decompositions [12], and similar principles have also been used recently to develop message-passing approximations for marginal MAP [13].

Our contributions. In this paper, we develop AND/OR branch and bound search for marginal MAP, using a heuristic created by extending weighted mini-bucket to the constrained elimination order of marginal MAP. We evaluate both a single-pass heuristic, which uses cost-shifting by moment matching (WMB-MM) during construction, and an iterative version that passes messages on the corresponding join-graph (WMB-JG). We show empirically that the new heuristic functions almost always improve over standard mini-bucket, and in many cases give tighter bounds and faster searches than the unconstrained join-tree meth-

ods, yielding far more empowered search algorithms.

We demonstrate the effectiveness of the proposed search algorithms against the two previous methods at solving a variety of problem instances derived from the recent PASCAL2 Inference Challenge benchmarks. Our results show not only orders of magnitude improvements over the current state-of-the-art approaches but also the ability to solve many instances that could not be solved before.

Following background and brief overview of earlier work (Sections 2 and 3), Section 4 presents the AND/OR search approach for marginal MAP. Section 5 describes the weighted mini-bucket schemes, Section 6 is dedicated to our empirical evaluation and Section 7 concludes.

2 BACKGROUND

A *graphical model* is a tuple $\mathcal{M} = \langle \mathbf{X}, \mathbf{D}, \mathbf{F} \rangle$, where $\mathbf{X} = \{X_i : i \in V\}$ is a set of variables indexed by set V and $\mathbf{D} = \{D_i : i \in V\}$ is the set of their finite domains of values. $\mathbf{F} = \{\psi_\alpha : \alpha \in F\}$ is a set of discrete positive real-valued local functions defined on subsets of variables, where we use $\alpha \subseteq V$ and $\mathbf{X}_\alpha \subseteq \mathbf{X}$ to indicate the *scope* of function ψ_α , ie, $\mathbf{X}_\alpha = \text{var}(\psi_\alpha) = \{X_i : i \in \alpha\}$. The function scopes imply a *primal graph* whose vertices are the variables and which includes an edge connecting any two variables that appear in the scope of the same function. The graphical model \mathcal{M} defines a factorized probability distribution on \mathbf{X} , $P(\mathbf{X}) = \frac{1}{Z} \prod_{\alpha \in F} \psi_\alpha$. The *partition function*, Z , normalizes the probability to sum to one.

Let \mathbf{X}_S be a subset of \mathbf{X} and $\mathbf{X}_M = \mathbf{X} \setminus \mathbf{X}_S$ be the complement of \mathbf{X}_S . The *Marginal MAP* problem is to find the assignment x_M^* to variables \mathbf{X}_M that maximizes the value of the marginal distribution after summing out variables \mathbf{X}_S :

$$x_M^* = \underset{\mathbf{X}_M}{\operatorname{argmax}} \sum_{\mathbf{X}_S} \prod_{\alpha \in F} \psi_\alpha \quad (1)$$

We call \mathbf{X}_M “MAP variables”, and \mathbf{X}_S “sum variables”.

If $\mathbf{X}_S = \emptyset$ then the problem is also known as maximum *a posteriori* (MAP) inference. The marginal MAP problem is however significantly more difficult. The decision problem for marginal MAP was shown to be NP^{PP} -complete [1], while the decision problem for MAP is only NP-complete [14]. The main difficulty arises because the max and sum operators in Eq. (1) do not commute, which restricts efficient elimination orders to those in which all sum variables \mathbf{X}_S are eliminated before any max variables \mathbf{X}_M .

Bucket Elimination (BE) [15] solves the marginal MAP problem exactly by eliminating the variables in sequence. Given a *constrained elimination order* ensuring the sum variables are processed before the max variables, BE partitions the functions into buckets, each associated with a single variable. A function is placed in the bucket of its

argument that appears latest in the ordering. BE processes each bucket, from last to first, by multiplying all functions in the current bucket and eliminating the bucket’s variable (by summation for sum variables and by maximization for MAP variables), resulting in a new function which is placed in an earlier bucket. The complexity of BE is time and space exponential in the *constrained induced width* w_c^* of the primal graph given a constrained elimination order [15]. BE can be viewed as message passing in a join-tree whose nodes correspond to buckets and which connects nodes a, b if the function generated by a ’s bucket is placed in b ’s [16].

Mini-Bucket Elimination (MBE) [3] is an approximation algorithm designed to avoid the space and time complexity of full bucket elimination by partitioning large buckets into smaller subsets, called *mini-buckets*, each containing at most i (called i -bound) distinct variables. The mini-buckets are processed separately [3]. MBE processes sum buckets and the max buckets differently. Max mini-buckets (in \mathbf{X}_M) are eliminated by maximization, while for variables in \mathbf{X}_S , one (arbitrarily selected) mini-bucket is eliminated by summation, while the rest of the mini-bucket are eliminated by maximization. MBE outputs an upper bound on the optimal marginal MAP value. The complexity of the algorithm, which is parametrized by the i -bound, is time and space exponential in i only. When i is large enough (i.e., $i > w_c^*$), MBE coincides with full BE. MBE is often used to generate heuristics for branch and bound search.

Another related approximation with bounded complexity, more similar in structure to join-tree inference, is *Mini-Cluster-Tree Elimination* (MCTE) [5]. In MCTE, we pass messages along the structure of the join-tree, except that when computing a message, rather than combining all the functions in the cluster, we first partition it into mini-clusters, such that each mini-cluster has a bounded number of variables (the i -bound). Each mini-cluster is then processed separately to compute a set of outgoing messages. Like MBE, this procedure produces an upper bound on the results of exact inference, and increasing i typically provides tighter bounds, but at higher computational cost. Thus, both MBE and MCTE allow the user to trade upper bound accuracy for time and space complexity.

3 CURRENT SEARCH METHODS

The current state-of-the-art methods for marginal MAP are based on branch and bound search using specialized heuristics. In particular, Park and Darwiche [6] construct an upper bound on each subproblem using a modified join-tree algorithm along an *unconstrained* elimination order that interleaves the MAP and sum variables. During search, the join-tree is fully re-evaluated at each node in order to compute upper bounds for all uninstantiated MAP variables simultaneously, which allows the use of dynamic variable ordering. Although this approach provides effective bounds,

Algorithm 1: BBBT for marginal MAP

Input: Graphical model $\mathcal{M} = \langle \mathbf{X}, \mathbf{D}, \mathbf{F} \rangle$, i -bound i , unassigned MAP variables \mathbf{X}_M , lower bound L , partial assignment to MAP variables \bar{x}

Output: Optimal marginal MAP value

```
1 if  $\mathbf{X}_M = \emptyset$  then
2    $\underline{\quad}$  return  $\text{Solve}(\mathcal{M}|\bar{x})$ ;
3 else
4    $X_k \leftarrow \text{SelectVar}(\mathbf{X}_M)$ ;
5   Update  $\text{MCTE}(i)$ ;
6   foreach  $\text{value } x_k \in D_k$  do
7     Assign  $X_k$  to  $x_k$ :  $\bar{x} \leftarrow \bar{x} \cup \{X_k = x_k\}$ ;
8      $U(\bar{x}) \leftarrow \text{extract}(\text{MCTE}(i))$ 
9     if  $U(\bar{x}) > L$  then
10       $\underline{\quad}$   $L = \max(L, \text{BBBT}(i, \mathbf{X}_M \setminus \{X_k\}, L, \bar{x}))$ ;
11       $\bar{x} \leftarrow \bar{x} \setminus \{X_k = x_k\}$ ;
12 return  $L$ ;
```

the computation can be quite expensive. More recently, Yuan and Hansen [7] proposed an incremental evaluation of the join-tree bounds which reduces significantly their computational overhead during search. However, this requires the search algorithm to follow a static variable ordering. In practice, Yuan and Hansen’s method proved to be cost effective, considerably outperforming [6]. However, both methods require that the induced width of the unconstrained join tree is small enough to be feasible, which often may not be the case.

3.1 ALGORITHM BBBT

Our first two algorithms, then, can be viewed as generalizations of [6] and [7] schemes for models with high unconstrained induced width. In particular, we use $\text{MCTE}(i)$ to approximate the exact, unconstrained join-tree inference to accommodate a maximum clique size defined by the i -bound i . The resulting branch and bound with $\text{MCTE}(i)$ heuristics, abbreviated hereafter by BBBT¹, for marginal MAP is given in Algorithm 1.

The algorithm is called initially as $\text{BBBT}(i, \mathbf{X}_M, 0, \emptyset)$, where \mathbf{X}_M are the MAP variables of the input graphical model, and i is the i -bound. The algorithm maintains the best solution found so far, giving a lower bound L on the optimal marginal MAP value. The algorithm searches the simple tree of all partial variable assignments (also called the OR tree). At each step, BBBT uses $\text{MCTE}(i)$ to compute an upper bound $U(\bar{x})$ on the optimal marginal MAP extension of the current partial MAP assignment \bar{x} (lines 5-8). If $U(\bar{x}) \leq L$, then the current assignment \bar{x} cannot lead to a better solution and the algorithm can backtrack (line 9). Otherwise, BBBT expands the current assignment by selecting the next MAP variable in a static or dynamic

¹For consistency with prior work, we use the name used in [17], (Branch and Bound with Bucket-Tree heuristic) to denote the same algorithm applied to pure MAP queries.

variable ordering (line 4) and recursively solves a set of subproblems, one for each un-pruned domain value. Notice that when \bar{x} is a complete assignment, BBBT calculates its marginal MAP value by solving a summation task over $\mathcal{M}|\bar{x}$, the subproblem defined by the sum variables conditioned on \bar{x} (line 2). Given sufficient resources (high enough i -bound), this can be done by variable elimination, but for consistency with our other algorithms, our implementation uses AND/OR search with caching [18] (see also Section 4). If a better new assignment is found then the lower bound L is updated (line 10).

If $\text{MCTE}(i)$ is fully re-evaluated at each iteration, it produces upper bounds for all uninstantiated MAP variables simultaneously. In this case, BBBT can accommodate dynamic variable orderings and can thus be viewed as a generalization of Park and Darwiche [6]. Alternatively, $\text{MCTE}(i)$ can be done in an incremental manner as in [7]. In this case BBBT requires a static variable ordering and can be viewed as a generalization of Yuan and Hansen.

4 AND/OR SEARCH

Significant improvements in search for pure MAP inference have been achieved by using AND/OR search spaces, which often capture problem structure far better than standard OR search methods [18]. In this section, we give an AND/OR search algorithm for marginal MAP. First, we define the *pseudo tree* of the primal graph, which defines the search space and captures problem decomposition.

DEFINITION 1 (pseudo tree) A pseudo tree of an undirected graph $G = (V, E)$ is a directed rooted tree $\mathcal{T} = (V, E')$ such that every arc of G not included in E' is a back-arc in \mathcal{T} , namely it connects a node in \mathcal{T} to one of its ancestors. The arcs in E' may not all be included in E .

The set of valid pseudo trees for marginal MAP is restricted to those for which the MAP variables form a *start pseudo tree*, a subgraph of pseudo tree \mathcal{T} that has the same root as \mathcal{T} . Given a graphical model $\mathcal{M} = \langle \mathbf{X}, \mathbf{D}, \mathbf{F} \rangle$ with primal graph G and pseudo tree \mathcal{T} of G , the *AND/OR search tree* $S_{\mathcal{T}}$ based on \mathcal{T} has alternating levels of OR nodes corresponding to the variables and AND nodes corresponding to the values of the OR parent’s variable, with edges weighted according to \mathbf{F} . Identical subproblems, identified by their *context* (the partial instantiation that separates the subproblem from the rest of the problem graph), can be merged, yielding an *AND/OR search graph* [18]. Merging all context-mergeable nodes yields the *context minimal AND/OR search graph*, denoted $C_{\mathcal{T}}$. The size of $C_{\mathcal{T}}$ is exponential in the induced width of G along a depth-first traversal of \mathcal{T} (i.e., the constrained induced width) [18].

A *solution tree* \hat{x} of $C_{\mathcal{T}}$ is a subtree that: (1) contains the root of $C_{\mathcal{T}}$; (2) if an internal OR node $n \in C_{\mathcal{T}}$ is in \hat{x} , then n is labeled by a MAP variable and exactly one of its

is not compatible, in general, with the constrained pseudo tree that drives the AOBB search order. For this reason, we next turn to improving our mini-bucket bounds.

5 MINI-BUCKET FOR MARGINAL MAP

In this section, we develop improved, constrained order mini-bucket bounds compatible with AOBB search. MBE has been effective for pure MAP, but less so for marginal MAP; previously, its bounds appeared to be far less accurate than the unconstrained join-tree bounds [6, 7]. Therefore, we revisit the mini-bucket approach and enhance it with recent iterative cost-shifting schemes [10, 9, 13].

5.1 WEIGHTED MINI-BUCKETS

Weighted mini-bucket elimination (WMB) [10] is a recent algorithm developed for likelihood (summation) tasks that replaces the naïve mini-bucket bound with Hölder’s inequality. For a given variable X_k , the mini-buckets Q_{kr} associated with X_k are assigned a non-negative *weight* $w_{kr} \geq 0$, such that $\sum_r w_{kr} = 1$. Then, each mini-bucket r is eliminated using a weighted or power sum, $(\sum_{X_k} f(X)^{1/w_{kr}})^{w_{kr}}$. It is useful to note that w_{kr} can be interpreted as a “temperature”; if $w_{kr} = 1$, it corresponds to a standard summation, while if $w_{kr} \rightarrow 0$, it instead corresponds to a maximization over X_k . Thus, standard mini-bucket corresponds to choosing one mini-bucket r with $w_{kr} = 1$, and the rest with weight zero.

Weighted mini-bucket is closely related to variational bounds on the likelihood, such as conditional entropy decompositions [12] and tree-reweighted belief propagation (TRBP) [11]. The single-pass algorithm of Liu and Ihler [10] mirrors standard mini-bucket, except that within each bucket a cost-shifting (or reparameterization) operator is performed, which matches the marginal beliefs (or “moments”) across mini-buckets to improve the bound.

The temperature viewpoint of the weights enables us to apply a similar procedure for marginal MAP. In particular, for $X_k \in \mathbf{X}_S$, we enforce $\sum_r w_{kr} = 1$, while for $X_k \in \mathbf{X}_M$, we take $\sum_r w_{kr} = 0$ (so that $w_{kr} = 0$ for all r). The resulting algorithm, listed in Algorithm 3, treats MAP and sum variables differently: for sum variables it mirrors [10], while taking the zero-temperature limit for MAP variables we obtain the max-marginal matching operations described for pure MAP problems in [9]. This mirrors the result of Weiss et al. [19], that the linear programming relaxation for MAP corresponds to a zero-temperature limit of TRBP.

5.2 ITERATIVE UPDATES

While the single-pass algorithm is often very effective, we can further improve it using iterative updates. The iterative weighted mini-bucket algorithm [10], alternates be-

Algorithm 3: WMB-MM(i)

Input: Graphical model $\mathcal{M} = \langle \mathbf{X}, \mathbf{D}, \mathbf{F} \rangle$, MAP variables \mathbf{X}_M , constrained ordering $o = X_1, \dots, X_n$, i -bound i
Output: Upper bound on optimal marginal MAP value

```

1 foreach  $k \leftarrow n$  downto 1 do
  // Create bucket  $\mathbf{B}_k$  and mini-buckets  $Q_{kr}$ 
2  $\mathbf{B}_k \leftarrow \{\psi_\alpha | \psi_\alpha \in \mathbf{F}, X_k \in \text{var}(\psi_\alpha)\}; \mathbf{F} \leftarrow \mathbf{F} \setminus \mathbf{B}_k;$ 
3 Let  $\mathcal{Q} = \{Q_{k1}, \dots, Q_{kR}\}$  be an  $i$ -partition of  $\mathbf{B}_k$ ;
4 foreach  $r = 1$  to  $R$  do
5    $\psi_{kr} = \prod_{\psi \in Q_{kr}} \psi; \mathbf{Y}_r = \text{vars}(Q_{kr}) \setminus X_k;$ 
  // Moment Matching
6 if  $X_k \in \mathbf{X}_S$  then
7   Assign mini-bucket  $r$  weight  $w_{kr} > 0$ , st  $\sum_r w_{kr} = 1$ ;
8    $\mu_r = \sum_{\mathbf{Y}_r} (\psi_{kr})^{1/w_{kr}}; \mu = \prod_r (\mu_r)^{w_{kr}};$ 
9   Update  $\psi_{kr} = \psi_{kr} \cdot \left(\frac{\mu}{\mu_r}\right)^{w_{kr}};$ 
10 else
11    $\mu_r = \max_{\mathbf{Y}_r} \psi_{kr}; \mu = \left(\prod_r \mu_r\right)^{1/R};$ 
12   Update  $\psi_{kr} = \psi_{kr} \cdot \left(\frac{\mu}{\mu_r}\right);$ 
  // Downward Messages (eliminate  $X_k$ )
13 foreach  $r = 1$  to  $R$  do
14   if  $X_k \in \mathbf{X}_S$  then  $\lambda_{kr} \leftarrow (\sum_{X_k} (\psi_{kr})^{1/w_{kr}})^{w_{kr}};$ 
15   else  $\lambda_{kr} \leftarrow \max_{X_k} \psi_{kr};$ 
16    $\mathbf{F} \leftarrow \mathbf{F} \cup \{\lambda_{kr}\};$ 
17 return  $\prod_{\psi \in \mathbf{F}} \psi$ 

```

tween downward passes, which look like standard mini-bucket with cost-shifting, and upward passes, which compute messages used to “focus” the cost shifting in the next downward pass. The algorithm can be viewed as message passing on a join graph defined by the mini-bucket cliques, and is listed in Algorithm 4.

Standard MBE computes “downward” messages $\lambda_{kr} = m_{a \rightarrow c}$ from each clique $a = (kr)$ (the r th mini-bucket for variable X_k) to a single child clique $c = \text{ch}(a)$. For the iterative version, we also compute “upward” messages $m_{c \rightarrow a}$ from clique c to its parent cliques $a \in \text{pa}(c)$. For $w_a > 0, w_c > 0$, these upward messages are given by [10]:

$$m_{c \rightarrow a} \propto \left[\sum_{Y_c \setminus Y_a} (\psi_c m_{\sim c})^{1/w_c} m_{a \rightarrow c}^{-1/w_a} \right]^{w_a}$$

where $\psi_c = \prod_{\psi \in Q_c} \psi$ are the model factors assigned to clique c , and $m_{\sim c}$ is the product of all messages into c .

These upward messages are used during the cost-shifting updates of X_k in later downward passes:

$$\forall r, \mu_{kr} \propto \sum_{Y_{kr}} (\psi_{kr} m_{\sim kr})^{1/w_{kr}}; \quad \mu = \left(\prod_r (\mu_{kr})^{w_{kr}} \right)^{1/w_k}$$

$$\forall r, \psi_{kr} \leftarrow \psi_{kr} \left(\frac{\mu}{\mu_{kr}} \right)^{\gamma w_{kr}}$$

in which we include the upward message $m_{\text{ch}(kr) \rightarrow kr}$ in the marginals μ_{kr} being matched, and define $w_k = \sum_r w_{kr}$. These fixed-point updates are not guaranteed to be monotonic; to assist convergence, we also include a “step size”

$\gamma \leq 1$. By initializing the upward messages $m_{\text{ch}(c) \rightarrow c} = 1$ and taking $\gamma = 1/t$, the first iteration of Alg. 4 corresponds exactly to WMB-MM (Alg. 3).

For marginal MAP, we can take the limit as some weights $w_a = \epsilon \rightarrow 0$; then, when both a and $c = \text{ch}(a)$ correspond to MAP variables we have

$$m_{c \rightarrow a} \propto \left[\max_{Y_c \setminus Y_a} (\psi_c m_{\sim c}) m_{a \rightarrow c}^{-1} \right]$$

$$\mu_a = \max_{Y_a} (\psi_a m_{\sim a}); \quad \mu = \left(\prod_a \mu_a \right)^{\frac{1}{|\mathcal{Q}|}}; \quad \psi_a \leftarrow \psi_a \left(\frac{\mu}{\mu_a} \right)^\gamma$$

When clique a corresponds to a sum variable and clique $c = \text{ch}(a)$ to a MAP variable, we take $w_c = \epsilon$ to give:

$$m_{c \rightarrow a} \propto \left[\sum_{Y_c \setminus Y_a} \sigma_\epsilon(\psi_c m_{\sim c}) m_{a \rightarrow c}^{-1/w_a} \right]^{w_a}$$

$$\sigma_\epsilon(f(X)) = (f(X) / \max_x f(x))^{1/\epsilon}$$

When $\epsilon \rightarrow 0$, σ_ϵ becomes an indicator function of the maximizing arguments of f , “focusing” the matching step at parent a on configurations relevant to the max values of child c . The resulting algorithm is also closely related to a (tree-reweighted) mixed-product belief propagation algorithm for marginal MAP [13]. Unfortunately, directly taking $\epsilon = 0$ can cause the objective function to be highly non-smooth, and lead to undesirable, non-monotonic fixed-point updates. To alleviate this, in practice we use a schedule $\epsilon = 1/t$ to decrease the temperature over iterations.

6 EXPERIMENTS

We empirically evaluate the proposed branch and bound algorithms on problem instances derived from benchmarks used in the PASCAL2 Inference Challenge [20] as well as the original instances from [7].

Algorithms. We consider three AND/OR branch and bound search algorithms (Section 4): AOBB guided by basic MBE(i) heuristics (denoted AOBB), AOBB guided by heuristics from WMB-MM(i) (denoted AOBB-MM), and AOBB guided by heuristics from WMB-JG(i) (denoted AOBB-JG), respectively. All of the mini-bucket heuristics were generated in a pre-processing phase, prior to search. The weighted schemes used uniform weights. In addition, we also tested two OR branch and bound schemes guided by MCTE(i) heuristics, denoted BBBTi and BBBTd, respectively. BBBTi performs MCTE(i) incrementally, while BBBTd fully re-evaluates MCTE(i) at each iteration.

We compare all five algorithms against each other and against the current state-of-the-art branch and bound with incremental join-tree upper bounds [7], denoted by YUAN, along with the original approach by Park and Darwiche [6], denoted by PARK. Algorithms BBBTd and PARK

Algorithm 4: WMB-JG(i)

Input: Graphical model $\mathcal{M} = \langle \mathbf{X}, \mathbf{D}, \mathbf{F} \rangle$, constrained ordering $o = X_1, \dots, X_n$, i -bound i , number of iterations T
Output: Upper bound on optimal marginal MAP value

```

1 for  $t = 1$  to  $T$  do
  // Downward pass with moment matching
2  foreach  $k \leftarrow n$  downto 1 do
3    Let  $\mathcal{Q} = \{Q_a | a = kr\}$  be the mini-buckets of  $\mathbf{B}_k$ ;
4    foreach  $Q_a \in \mathcal{Q}$  do
5       $\psi_a = \prod_{\psi \in Q_a} \psi$ ;
6       $\mathbf{Y}_a = \text{vars}(Q_a) \setminus X_k$ ;
7       $m_{\sim a} = m_{\text{ch}(a) \rightarrow a} \cdot \prod_{p \in \text{pa}(a)} m_{p \rightarrow a}$ ;
8    if  $X_k \in \mathbf{X}_S$  then
9      foreach  $Q_a \in \mathcal{Q}$  do  $\mu_a = \sum_{\mathbf{Y}_a} (\psi_a m_{\sim a})^{1/w_a}$ ;
10      $\mu = \prod_{Q_a \in \mathcal{Q}} (\mu_a)^{w_a}$ ;
11     foreach  $Q_a \in \mathcal{Q}$  do Update  $\psi_a = \psi_a \cdot (\mu / \mu_a)^{w_a}$ ;
12   else
13     foreach  $Q_a \in \mathcal{Q}$  do  $\mu_a = \max_{\mathbf{Y}_a} (\psi_a m_{\sim a})$ ;
14      $\mu = \prod_{Q_a \in \mathcal{Q}} (\mu_a)^{1/|\mathcal{Q}|}$ ;
15     foreach  $Q_a \in \mathcal{Q}$  do Update  $\psi_a = \psi_a \cdot (\mu / \mu_a)$ ;
16   foreach  $Q_a \in \mathcal{Q}, c = \text{ch}(a), \text{ do}$ 
17     if  $X_k \in \mathbf{X}_S$  then
18        $m_{a \rightarrow c} = (\sum_{X_k} (\psi_a m_a)^{1/w_a})^{w_a}$ ;
19     else
20        $m_{a \rightarrow c} = \max_{X_k} (\psi_a m_a)$ ;
21   // Backward pass
22   foreach  $k \leftarrow 1$  to  $n$  do
23     Let  $\mathcal{Q} = \{Q_c | c = kr\}$  be the mini-buckets of  $\mathbf{B}_k$ ;
24     foreach  $Q_c \in \mathcal{Q}$  and  $a \in \text{pa}(c)$ , with  $c = kr, a = js$  do
25        $\mathbf{Y} = \text{vars}(Q_c) \setminus \text{vars}(Q_a)$ ;
26       if  $X_k \in \mathbf{X}_S$  and  $X_j \in \mathbf{X}_S$  then
27          $m_{c \rightarrow a} =$ 
28          $(\sum_{\mathbf{Y}} (\psi_c m_{\sim c})^{1/w_c} \cdot (m_{a \rightarrow c})^{-1/w_a})^{w_a}$ ;
29       if  $X_k \in \mathbf{X}_M$  and  $X_j \in \mathbf{X}_M$  then
30          $m_{c \rightarrow a} = (\max_{\mathbf{Y}} (\psi_c m_{\sim c}) \cdot (m_{a \rightarrow c})^{-1})^{w_a}$ ;
31       if  $X_k \in \mathbf{X}_S$  and  $X_j \in \mathbf{X}_M$  then
32          $m_{c \rightarrow a} =$ 
33          $(\sum_{\mathbf{Y}} \sigma_\epsilon(\psi_c m_{\sim c}) \cdot (m_{a \rightarrow c})^{-1/w_a})^{w_a}$ ;
34   return upper bound from  $\mathbf{B}_1$ ;
```

use a dynamic variable ordering and select the next MAP variable whose domain values have the most asymmetric bounds. Algorithms BBBTi and YUAN are restricted to a static variable ordering that corresponds to a post-order traversal of the underlying join-tree. The pseudo trees guiding the AND/OR algorithms were obtained by a modified min-fill heuristic [18] that constrained the MAP variables to form a start pseudo tree. All algorithms were implemented in C++ (64-bit) and the experiments were run on a 2.6GHz 8-core processor with 80 GB of RAM.

Benchmarks. Our problem instances were derived from three PASCAL2 benchmarks: *segbin* (image segmentation), *protein* (protein side-chain interaction) and *promedas* (medical diagnosis expert system). For each

Table 1: Upper bounds (log scale) and CPU time (sec) for a typical set of instances. $i = 10$ and $i = 20$.

instance (n, m, k, w_c^*, w_u^*)	i	MBE UB/time	WMB-MM UB/time	MCTE UB/time	5 iterations UB/time	WMB-JG 10 iterations UB/time	100 iterations UB/time	JT UB/time
cpcs360 (360,25,2,24,20)	10	5.9607/0.03	-0.0228/0.04	4.3008/0.16	-0.0353/0.34	-0.0360/0.75	-0.0363/6.71	-0.0468/4.73
	20	2.4871/11.4	-0.0402/1.36	-0.0468/3.45	-0.0465/47.2	-0.0467/145	-0.0468 /1339	
2-17-s-s (228,69,2,20,15)	10	-44.2658/0.02	-49.5830/0.01	-40.3520/0.06	-55.4555/0.12	-55.5996/0.24	-55.6633/2.44	-55.5170/0.31
	20	-55.5083/3.54	-55.5082/0.30	-55.5170/0.36	-55.7433/5.38	-55.7436/12.7	-55.7437 /197	
or-chain-10.fg-s (453,135,2,22,18)	10	-10.5621/0.01	-13.2118/0.01	-6.4940/0.1	-17.2899/0.10	-18.7859/0.18	-21.3428/1.71	-21.0314/4.29
	20	-18.2977/3.56	-19.3815/0.33	-9.8054/0.5	-21.3600 /5.46	-21.3600 /11.8	-21.3600 /137	
cpcs422 (422,74,2,74,23)	10	10.026/0.61	-1.3206/0.88	7.0553/1.62	-1.3764/4.06	-1.3878/8.67	-1.4275/75.1	-1.4982 /41.6
	20	7.9245/18.5	-1.4427/9.29	-0.1331/9.78	-1.4545/191	-1.4554/371	-1.4718/2353	
2-2-s-l (227,68,2,73,14)	10	-57.0433/0.04	-75.1643/0.03	-39.1568/0.07	-80.9224/0.17	-81.5811/0.33	-81.9044/3.56	-81.5883/0.14
	20	-67.2268/5.52	-80.4492/1.68	-81.5883/0.17	-82.0108/25.0	-82.1039/58.6	-82.1960 /400	
or-chain-18.fg-l (890,267,2,25,8)	10	-2.7317/0.01	-2.5168/0.02	-2.3325/0.42	-6.4279/0.36	-7.1655/0.51	-11.1244/3.78	-11.4487/0.43
	20	-10.2463/0.88	-11.4534 /0.07	-11.4487/0.46	-11.4534 /1.48	-11.4534 /2.97	-11.4534 /31.7	

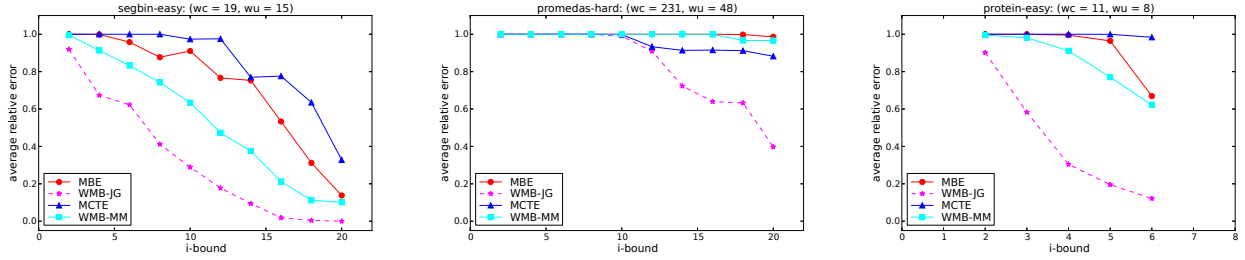


Figure 3: Average relative error (w.r.t. tightest upper bound) as a function of i -bound. WMB-JG(i) ran for 10 iterations.

network, we generated two marginal MAP problem instances with m MAP variables, as follows: an *easy* instance such that the MAP variables were selected as the first m variables from a breadth-first traversal of a pseudo tree obtained from a hypergraph decomposition of the primal graph (ties were broken randomly) [8], and a *hard* instance where the MAP variables were selected uniformly at random. The *easy* instances were designed such that problem decomposition is maximized and the constrained and unconstrained elimination orders are relatively close to each other, thus having comparable induced widths. In contrast, the *hard* instances tend to have very large constrained induced widths. We selected 30% of the variables as MAP variables. In total we evaluated 120 problem instances (20 easy and 20 hard instances per benchmark).

In all experiments we report total CPU time in seconds and number of nodes visited during search. We also record the problem parameters: number of variables (n), max domain size (k), number of MAP variables (m), and the constrained (w_c^*) and unconstrained (w_u^*) induced widths. The best performance points are highlighted. In each table, 'oom' stands for out-of-memory, while '-' denotes out-of-time.

Results: quality of the upper bounds. We compare the accuracy of the upper bounds obtained by the mini-bucket schemes MBE(i), WMB-MM(i) and WMB-JG(i) against those produced by the unconstrained join-tree scheme, denoted JT, and its generalization MCTE(i).

Table 1 shows results on a typical set of problem instances from both *easy* (top 3) and *hard* (bottom 3) categories, for two values of i -bound: $i = 10, 20$. For every problem instance, for each algorithm we report the upper bound ob-

tained (lower values are better) and CPU time in seconds. The iterative scheme WMB-JG(i) ran for 5, 10 or 100 iterations, respectively. We see clearly that for all instances WMB-MM(i) provides significantly tighter upper bounds than the corresponding pure MBE(i) in a comparable CPU time (see also Figure 3). On the other hand, WMB-JG(i) is able to converge to the most accurate bounds in 4 out of 6 cases, but at a much higher computational cost. JT bounds are typically tighter than those produced by MCTE(i) and MBE(i) which is consistent with previous studies [6, 7].

In Figure 3 we plot the average relative error with respect to the tightest upper bound obtained, as a function of the i -bound. Since, the JT bounds were available only on a relatively small fraction of the instances tested, they are omitted for clarity. We observe that if given enough time WMB-JG(i) is superior to all its competitors, especially for larger i -bounds. However, if time is bounded, then WMB-MM(i) provides a cost-effective alternative. Notice also that when the gap between the constrained and unconstrained induced width is very large, then MCTE(i) provides more accurate bounds than MBE(i) and WMB-MM(i) (eg, promedas hard), because MCTE(i) does less partitioning in this case. When the gap is relatively small, then the mini-bucket based bounds are often superior to the MCTE(i) ones for the same i -bound (eg, segbin easy).

Results: comparison with state-of-the-art search. Tables 2 and 3 report CPU time in seconds and number of nodes expanded by each search algorithm on a subset of instances from the protein and promedas benchmarks. The columns are indexed by the i -bound and the time limit was set to 1 hour. WMB-JG(i) ran for 10 iterations. We can

Table 2: CPU time (sec) and nodes for the protein instances. Time limit 1 hour. WMB-JG(i) ran for 10 iterations.

instance	algorithm	$i = 2$		$i = 3$		$i = 4$		$i = 5$		$i = 6$		YUAN PARK	
(n, m, kw_c^*, w_u^*)		time	nodes	time	nodes	time	nodes	time	nodes	time	nodes	time	nodes
protein easy instances													
pdb1a1x (95,28,81,14,14)	AOBB	-	-	-	-	-	-	-	-	-	-	-	-
	AOBB-JG	539	1746192	85	314801	164	3415	3067	746	-	-	-	oom
	AOBB-MM	-	-	-	-	601	7625110	709	9004715	2087	316563	-	oom
	BBBTd	-	-	-	-	-	-	-	-	-	-	-	-
	BBBTi	-	-	-	-	-	-	-	-	-	-	-	-
pdb1a62 (105,31,81,13,10)	AOBB	-	-	1533	12650401	379	1505951	228	169618	753	274565	-	-
	AOBB-JG	-	-	13	35	62	35	523	35	2228	35	-	oom
	AOBB-MM	697	2437932	169	359560	114	135525	138	181286	112	1107	-	oom
	BBBTd	-	-	-	-	-	-	-	-	-	-	-	-
	BBBTi	-	-	-	-	-	-	-	-	-	-	-	-
pdb1ad2 (177,53,81,12,9)	AOBB	-	-	-	-	-	-	-	-	-	-	-	-
	AOBB-JG	-	-	76	1355	227	431	3368	424	-	-	-	oom
	AOBB-MM	-	-	-	-	983	838218	211	13902	-	-	-	oom
	BBBTd	-	-	-	-	-	-	-	-	-	-	-	-
	BBBTi	-	-	-	-	-	-	-	-	-	-	-	-
pdb1aho (54,16,81,7,6)	AOBB	61	119726	9	5483	4	735	21	283	154	48	-	-
	AOBB-JG	6	6581	4	365	19	271	65	17	1251	17	299	55
	AOBB-MM	49	19890	10	3274	8	2057	7	593	44	17	963	16
	BBBTd	7	1224	6	128	28	26	165	29	426	17	-	-
	BBBTi	77	291321	949	1151691	345	35506	-	-	356	4679	-	-

Table 3: CPU time (sec) and nodes for the promedas instances. Time limit 1 hour. WMB-JG(i) ran for 10 iterations.

instance	algorithm	$i = 4$		$i = 6$		$i = 10$		$i = 14$		$i = 18$		$i = 20$		PARK YUAN	
(n, m, k, w_c^*, w_u^*)		time	nodes	time	nodes	time	nodes	time	nodes	time	nodes	time	nodes	time	nodes
promedas easy instances															
or-chain-4.fg-e (691,207,2,33,26)	AOBB	-	-	-	-	65	6242529	14	1871710	4	471708	7	235860	-	-
	AOBB-JG	-	1046	75598793	-	9	1045873	55	5457626	6	208	19	1144	-	oom
	AOBB-MM	-	-	-	-	116	7354956	8	991915	1	156030	1	73526	-	oom
	BBBTd	-	-	-	-	579	39989	132	4624	233	1900	425	1285	-	-
	BBBTi	-	-	-	-	-	-	394	2001912	-	-	-	-	-	-
or-chain-17.fg-e (531,159,2,20,18)	AOBB	447	67968093	64	12082065	3	518292	1	162224	1	1920	2	0	87	159
	AOBB-JG	38	3943341	57	8830508	0	72575	0	6940	3	160	6	160	3	162
	AOBB-MM	238	26609470	65	9743803	2	306313	0	45462	0	757	0	521	-	-
	BBBTd	-	-	-	-	103	5520	85	2921	125	1363	148	633	-	-
	BBBTi	-	-	-	-	-	-	5	61467	10	29232	12	25588	-	-
or-chain-22.fg-e (1044,313,2,72,59)	AOBB	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	AOBB-JG	-	-	-	-	-	-	2118	183274481	-	-	-	-	-	oom
	AOBB-MM	-	-	-	-	-	-	-	-	-	-	-	-	-	oom
	BBBTd	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	BBBTi	-	-	-	-	-	-	-	-	-	-	-	-	-	-
promedas hard instances															
or-chain-4.fg-h (691,207,2,140,28)	AOBB	-	-	-	-	-	-	-	-	2254	124886725	-	-	-	-
	AOBB-JG	-	-	-	-	192	5529085	11	555059	21	377992	66	215655	-	oom
	AOBB-MM	-	-	-	-	752	17706171	304	13152476	188	5662611	78	2134464	-	oom
	BBBTd	-	-	-	-	-	-	-	-	1810	12397	-	-	-	-
	BBBTi	-	-	-	-	-	-	-	-	-	-	-	-	-	-
or-chain-8.fg-h (1195,358,2,255,39)	AOBB	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	AOBB-JG	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	AOBB-MM	-	-	-	-	-	-	-	-	-	-	1786	31316917	-	oom
	BBBTd	-	-	-	-	-	-	-	-	-	-	-	-	-	oom
	BBBTi	-	-	-	-	-	-	-	-	-	-	-	-	-	-
or-chain-17.fg-h (531,159,2,72,18)	AOBB	-	-	-	-	67	7544343	12	1282228	13	1556793	11	606211	-	-
	AOBB-JG	-	-	-	-	42	3992210	3	212839	8	230955	29	169192	259	159
	AOBB-MM	-	-	-	-	-	-	-	-	7	793696	287	9274776	4	439
	BBBTd	-	-	-	-	-	-	412	12954	861	6003	1931	4649	-	-
	BBBTi	-	-	-	-	477	5618175	61	494659	54	136679	106	126093	-	-

see clearly that AOBB-JG(i) is the overall best performing algorithm, especially for relatively small i -bounds. For example, on the pdb1a62, AOBB-JG(3) proves optimality in 13 seconds while AOBB(3) and AOBB-MM(3) finish in 1533 and 169 seconds, respectively. The search space explored by AOBB-JG(3) is also dramatically smaller than those explored by AOBB(3) or AOBB-MM(3). Therefore, the much stronger heuristics generated by WMB-JG(i) translate into impressive time savings. When the i -bound increases, the accuracy ceases to offset the computational overhead and the running time of AOBB-JG(i) increases (e.g., pdb1aho). In this case, AOBB-MM(i) is a cost-effective alternative, with reduced overhead for pre-compiling the heuristic (see also Figure 4 for a profile of the

CPU time of the algorithms across all benchmarks). The performance of algorithms YUAN and PARK is quite poor in this domain due to the relatively large unconstrained induced widths, which prevent computation of their heuristic. In contrast, BBBTi/BBBTd with relatively higher i -bounds are sometimes competitive and are able to solve more problem instances than YUAN/PARK.

For completeness, we also tested on the Bayesian networks from [7] (results omitted for space). We observed that all of our proposed algorithms were competitive with YUAN/PARK, but due to the relatively small unconstrained induced widths on these problems, very accurate join-tree heuristics could be computed. Thus, there was very little room for improvement by the new methods.

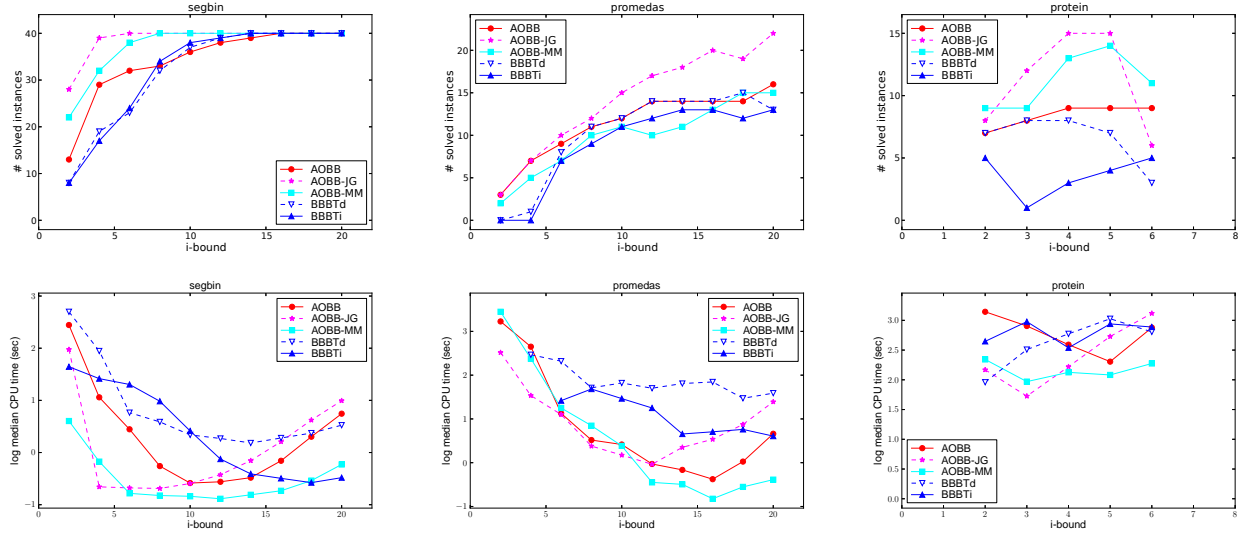


Figure 4: Number of instances solved (top) and median CPU time (bottom) as a function of i -bound for the segbin, promedas and protein instances. Time limit 1 hour. WMB-JG(i) ran for 10 iterations.

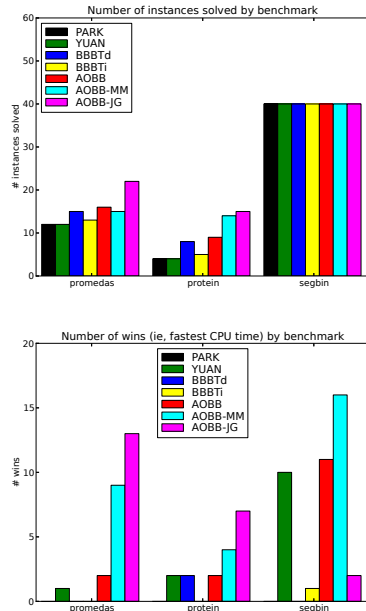


Figure 5: Number of instances solved (top) and number of wins (bottom) by benchmark.

Summary of the experiments. Figure 4 plots the number of problem instances solved from each benchmark (top) and the median CPU time (bottom) as a function of the i -bound. Clearly, AOBB-JG solves the largest number of instances across i -bounds. Moreover, the running time profile shows that AOBB-JG is faster at lower i -bounds due to more accurate heuristics, while AOBB-MM is faster at higher i -bounds due to reduced overhead. Figure 5 summarizes the total number of instances solved as well as the total number of wins (defining a ‘win’ as the fastest time) across the benchmarks, for all competing algorithms. Over-

all, we see that the proposed search algorithms consistently solve more problems and in many cases are significantly faster than the current approaches.

In summary, based on our empirical evaluation, we can conclude that:

- Cost-shifting (especially the iterative version) tightened significantly the MBE bounds for marginal MAP. This yielded considerably faster AOBB search.
- The AOBB algorithms with improved mini-bucket heuristics outperformed in many cases the previous search methods guided by join-tree based heuristics.

7 CONCLUSION

In this paper, we develop AND/OR branch and bound search algorithms for marginal MAP that use heuristics extracted from weighted mini-buckets with cost-shifting. We evaluate both a single-pass version of the heuristic with cost-shifting by moment matching as well as an iterative version that passes messages on the corresponding join-graph. We demonstrate the effectiveness of our proposed search algorithms against previous unconstrained join-tree based methods, which we also extend to apply to high induced-width models, through extensive empirical evaluations on a variety of benchmarks. Our results show not only orders of magnitude improvements over the current state-of-the-art, but also the ability to solve many instances that could not be solved before.

Acknowledgments This work was sponsored in part by NSF grants IIS-1065618 and IIS-1254071, and by the United States Air Force under Contract No. FA8750-14-C-0011 under the DARPA PPAML program.

References

- [1] J. Park. MAP complexity results and approximation methods. In *Uncertainty in Artificial Intelligence (UAI)*, pages 388–396, 2002.
- [2] D. Roth. On the hardness of approximate reasoning. *Artificial Intelligence*, 82(1-2):273–302, 1996.
- [3] R. Dechter and I. Rish. Mini-buckets: A general scheme of approximating inference. *Journal of ACM*, 50(2):107–153, 2003.
- [4] R. Mateescu, K. Kask, V. Gogate, and R. Dechter. Join-graph propagation algorithms. *Journal of Artificial Intelligence Research*, 37:279–328, 2010.
- [5] R. Dechter, K. Kask, and J. Larrosa. A general scheme for multiple lower bound computation in constraint optimization. In *Principles and Practice of Constraint Programming*, pages 346–360, 2001.
- [6] J. Park and A. Darwiche. Solving MAP exactly using systematic search. In *Uncertainty in Artificial Intelligence (UAI)*, pages 459–468, 2003.
- [7] C. Yuan and E. Hansen. Efficient computation of join-tree bounds for systematic MAP search. In *International Joint Conference on Artificial Intelligence (IJ-CAI)*, pages 1982–1989, 2009.
- [8] R. Marinescu and R. Dechter. AND/OR branch-and-bound search for combinatorial optimization in graphical models. *Artificial Intelligence*, 173(16-17):1457–1491, 2009.
- [9] A. Ihler, N. Flerova, R. Dechter, and L. Otten. Join-graph based cost-shifting schemes. In *Uncertainty in Artificial Intelligence (UAI)*, pages 397–406, 2012.
- [10] Q. Liu and A. Ihler. Bounding the partition function using hölder’s inequality. In *International Conference on Machine Learning (ICML)*, pages 849–856, 2011.
- [11] M. Wainwright, T. Jaakkola, and A. Willsky. A new class of upper bounds on the log partition function. *IEEE Trans. Info. Theory*, 51(7):2313–2335, July 2005.
- [12] A. Globerson and T. Jaakkola. Approximate inference using conditional entropy decompositions. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 130–138, 2007.
- [13] Q. Liu and A. Ihler. Variational algorithms for marginal MAP. *Journal of Machine Learning Research*, 14:3165–3200, 2013.
- [14] S. E. Shimony. Finding MAPs for belief networks is NP-hard. *Artificial Intelligence*, 2(68):399–410, 1994.
- [15] R. Dechter. Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, 113(1-2):41–85, 1999.
- [16] K. Kask, R. Dechter, J. Larrosa, and A. Dechter. Unifying cluster-tree decompositions for reasoning in graphical models. *Artificial Intelligence*, 166 (1-2):165–193, 2005.
- [17] R. Marinescu, K. Kask, and R. Dechter. Systematic vs non-systematic algorithms for solving the MPE task. In *Uncertainty in Artificial Intelligence (UAI)*, pages 394–402, 2003.
- [18] R. Dechter and R. Mateescu. AND/OR search spaces for graphical models. *Artificial Intelligence*, 171(2-3):73–106, 2007.
- [19] Y. Weiss, C. Yanover, and T. Meltzer. MAP estimation, linear programming and belief propagation with convex free energies. In *Uncertainty in Artificial Intelligence (UAI)*, pages 416–425, 2007.
- [20] G. Elidan, A. Globerson, and U. Heinemann. PASCAL 2011 probabilistic inference challenge. <http://www.cs.huji.ac.il/project/PASCAL/>, 2012.

Stochastic Discriminative EM

Andrés R. Masegosa^{†‡}

[†] Dept. of Computer and Information Science
Norwegian University of Science and Technology
Trondheim, Norway

[‡] Dept. of Computer Science and A. I.
University of Granada
Granada, Spain

Abstract

Stochastic discriminative EM (sdEM) is an online-EM-type algorithm for discriminative training of probabilistic generative models belonging to the natural exponential family. In this work, we introduce and justify this algorithm as a stochastic *natural gradient* descent method, i.e. a method which accounts for the information geometry in the parameter space of the statistical model. We show how this learning algorithm can be used to train probabilistic generative models by minimizing different discriminative loss functions, such as the negative conditional log-likelihood and the Hinge loss. The resulting models trained by sdEM are always generative (i.e. they define a joint probability distribution) and, in consequence, allows to deal with missing data and latent variables in a principled way either when being learned or when making predictions. The performance of this method is illustrated by several text classification problems for which a multinomial naive Bayes and a latent Dirichlet allocation based classifier are learned using different discriminative loss functions.

1 INTRODUCTION

Online learning methods based on stochastic approximation theory [19] have been a promising research direction to tackle the learning problems of the so-called Big Data era [1, 10, 12]. Stochastic gradient descent (SGD) is probably the best known example of this kind of techniques, used to solve a wide range of learning problems [9]. This algorithm and other versions [27] are usually employed to train discriminative models such as logistic regression or SVM [10].

There also are some successful examples of the use of SGD for discriminative training of probabilistic generative models, as is the case of deep belief networks [18]. However,

this learning algorithm cannot be used directly for the discriminative training of general generative models. One of the main reasons is that statistical estimation or risk minimization problems of generative models involve the solution of an optimization problem with a large number of *normalization constraints* [24], i.e. those which guarantee that the optimized parameter set defines a valid probabilistic model. Although successful solutions to this problem have been proposed [16, 20, 24, 31], they are based on ad-hoc methods which cannot be easily extended to other statistical models, and hardly scale to large data sets.

Stochastic approximation theory [19] has also been used for maximum likelihood estimation (MLE) of probabilistic generative models with latent variables, as is the case of the online EM algorithm [13, 29]. This method provides efficient MLE estimation for a broad class of statistical models (i.e. exponential family models) by sequentially updating the so-called *expectation parameters*. The advantage of this approach is that the resulting iterative optimization algorithm is fairly simple and amenable, as it does not involve any normalization constraints.

In this paper we show that the derivation of Sato's online EM [29] can be extended for the discriminative learning of generative models by introducing a novel interpretation of this algorithm as a natural gradient algorithm [3]. The resulting algorithm, called stochastic discriminative EM (sdEM), is an online-EM-type algorithm that can train generative probabilistic models belonging to the natural exponential family using a wide range of discriminative loss functions, such as the negative conditional log-likelihood or the Hinge loss. In opposite to other discriminative learning approaches [24], models trained by sdEM can deal with missing data and latent variables in a principled way either when being learned or when making predictions, because at any moment they always define a joint probability distribution. sdEM could be used for learning using large scale data sets due to its stochastic approximation nature and, as we will show, because it allows to compute the *natural gradient* of the loss function with no extra cost [3]. Moreover, if allowed by the generative model and the discriminative

loss function, the presented algorithm could potentially be used interchangeably for classification or regression or any other prediction task. But in this initial work, sdEM is only experimentally evaluated in classification problems.

The rest of this paper is organized as follows. Section 2 provides the preliminaries for the description of the sdEM algorithm, which is detailed in Section 3. A brief experimental evaluation is given in Section 4, while Section 5 contains the main conclusions of this work.

2 PRELIMINARIES

2.1 MODEL AND ASSUMPTIONS

We consider generative statistical models for prediction tasks, where Y denotes the random variable (or the vector-value random variable) to be predicted, X denotes the predictive variables, and y^* denotes a prediction, which is made according to $y^* = \arg \max_y p(y, x|\theta)$.

Assumption 1. *The generative data model belongs to a natural exponential family*

$$p(y, x|\theta) \propto \exp(\langle s(y, x), \theta \rangle - A_l(\theta))$$

where θ is the so-called natural parameter which belongs to the so-called natural parameter space $\Theta \in \mathbb{R}^K$, $s(y, x)$ is the vector of sufficient statistics belonging to a convex set $\mathcal{S} \subseteq \mathbb{R}^K$, $\langle \cdot, \cdot \rangle$ denotes the dot product and A_l is the log partition function.

Assumption 2. *We are given a conjugate prior distribution $p(\theta|\alpha)$ of the generative data model*

$$p(\theta|\alpha) \propto \exp(\langle s(\theta), \alpha \rangle - A_g(\alpha))$$

where the sufficient statistics are $s(\theta) = (\theta, -A_l(\theta))$ and the hyperparameter α has two components $(\bar{\alpha}, \nu)$. ν is a positive scalar and $\bar{\alpha}$ is a vector also belonging to \mathcal{S} [6].

2.2 DUAL PARAMETERIZATION AND ASSUMPTIONS

The so-called *expectation parameter* $\mu \in \mathcal{S}$ can also be used to parameterize probability distributions of the natural exponential family. It is a dual set of the model parameter θ [2]. This *expectation parameter* μ is defined as the expected vector of sufficient statistics with respect to θ :

$$\begin{aligned} \mu &\triangleq E[s(y, x)|\theta] = \int s(y, x)p(y, x|\theta)dydx \\ &= \partial A_l(\theta)/\partial \theta \end{aligned} \quad (1)$$

The transformation between θ and μ is one-to-one: μ is a dual set of the model parameter θ [2]. Therefore, Equation (1) can be inverted as: $\theta = \theta(\mu)$. That is to say, for each $\theta \in \Theta$ we always have an associated $\mu \in \mathcal{S}$ and both parameterize the same probability distribution.

For obtaining the *natural parameter* θ associated to an *expectation parameter* μ , we need to make use of the negative of the entropy,

$$\begin{aligned} H(\mu) &\triangleq \int p(y, x|\theta(\mu)) \ln p(y, x|\theta(\mu)) dydx \\ &= \sup_{\theta \in \Theta} \langle \mu, \theta \rangle - A_l(\theta) \end{aligned} \quad (2)$$

Using the above function, the natural parameter θ can be explicitly expressed as

$$\theta = \theta(\mu) = \partial H(\mu)/\partial \mu \quad (3)$$

Equations (1), (2), (3) define the Legendre-Fenchel transform.

Another key requirement of our approach is that it should be possible to compute the transformation from μ to θ in closed form:

Assumption 3. *The transformation from the expectation parameter μ to the natural parameter θ , which can be expressed as*

$$\theta(\mu) = \arg \max_{\theta \in \Theta} \langle \mu, \theta \rangle - A_l(\theta) \quad (4)$$

is available in closed form.

The above equation is also known as the *maximum likelihood function*, because $\theta(\frac{1}{n} \sum_{i=1}^n s(y_i, x_i))$ gives the maximum likelihood estimation θ^* for a data set with n observations $\{(y_1, x_1), \dots, (y_n, x_n)\}$.

For later convenience, we show the following relations between the Fisher Information matrices $I(\theta)$ and $I(\mu)$ for the probability distributions $p(y, x|\theta)$ and $p(y, x|\theta(\mu))$, respectively [23]:

$$I(\theta) = \frac{\partial^2 A_l(\theta)}{\partial \theta \partial \theta} = \frac{\partial \mu}{\partial \theta} = I(\mu)^{-1} \quad (5)$$

$$I(\mu) = \frac{\partial^2 H(\mu)}{\partial \mu \partial \mu} = \frac{\partial \theta}{\partial \mu} = I(\theta)^{-1} \quad (6)$$

2.3 THE NATURAL GRADIENT

Let $\mathcal{W} = \{w \in \mathbb{R}^K\}$ be a parameter space on which the function $L(w)$ is defined. When \mathcal{W} is a Euclidean space with an orthonormal coordinate system, the negative gradient points in the direction of steepest descent. That is, the negative gradient $-\partial L(w)/\partial w$ points in the same direction as the solution to:

$$\arg \min_{dw} L(w + dw) \text{ subject to } \|dw\|^2 = \epsilon^2 \quad (7)$$

for sufficiently small ϵ , where $\|dw\|^2$ is the squared length of a small increment vector dw connecting w and $w + dw$. This justifies the use of the classical gradient descent method for finding the minimum of $L(w)$ by taking steps (of size ρ) in the direction of the negative gradient:

$$w_{t+1} = w_t - \rho \frac{\partial L(w_t)}{\partial w} \quad (8)$$

However, when \mathcal{W} is a Riemannian space [4], there are no orthonormal linear coordinates, and the squared length of vector dw is defined by the following equation,

$$||dw||^2 = \sum_{ij} g_{ij}(w) dw_i dw_j \quad (9)$$

where the $K \times K$ matrix $G = (g_{ij})$ is called the Riemannian metric tensor, and it generally depends on w . G reduces to the identity matrix in the case of the Euclidean space [4].

In a Riemannian space, the steepest descent direction is not anymore the traditional gradient. That is, $-\partial L(w)/\partial w$ is not the solution of Equation (7) when the squared length of the distance of dw is defined by Equation (9). Amari [3] shows that this solution can be computed by pre-multiplying the traditional gradient by the inverse of the Riemannian metric G^{-1} ,

Theorem 1. *The steepest descent direction or the natural gradient of $L(w)$ in a Riemannian space is given by*

$$-\frac{\tilde{\partial} L(w)}{\tilde{\partial} w} = -G^{-1}(w) \frac{\partial L(w)}{\partial w} \quad (10)$$

where $\tilde{\partial} L(w)/\tilde{\partial} w$ denotes the *natural gradient*.

As argued in [3], in statistical estimation problems we should use gradient descent methods which account for the natural gradient of the parameter space, as the parameter space of a statistical model (belonging to the exponential family or not) is a Riemannian space with the Fisher information matrix of the statistical model $I(w)$ as the tensor metric [2], and this is the only invariant metric that must be given to the statistical model [2].

2.4 SATO'S ONLINE EM ALGORITHM

Sato's online EM algorithm [29] is used for maximum likelihood estimation of missing data-type statistical models. The model defines a probability distribution over two random or vector-valued variables X and Z , and is assumed to belong to the natural exponential family:

$$p(z, x|\theta) \propto \exp(\langle s(z, x), \theta \rangle - A_t(\theta))$$

where (z, x) denotes a so-called *complete data* event. The key aspect is that we can only observe x , since z is an unobservable event. In consequence, the loss function $\ell(x, \theta)$ ¹ is defined by marginalizing z : $\ell(x, \theta) = -\ln \int p(z, x) dz$.

The online setting assumes the observation of a non-finite data sequence $\{(x_t)\}_{t \geq 0}$ independently drawn according to the unknown data distribution π . The objective function that EM seeks to minimize is given by the following expectation: $L(\theta) = E[\ell(x, \theta)|\pi]$.

¹We derive this algorithm in terms of minimization of a loss function to highlight its connection with sdEM.

Sato [29] derived the stochastic updating equation of online EM by relying on the free energy formulation, or lower bound maximization, of the EM algorithm [22] and on a discounting averaging method. Using our own notation, this updating equation is expressed as follows,

$$\begin{aligned} \mu_{t+1} &= (1 - \rho_t)\mu_t + \rho_t E_z[s(z, x_t|\theta(\mu_t))] \\ &= \mu_t + \rho_t (E_z[s(z, x_t|\theta(\mu_t))] - \mu_t) \\ &= \mu_t + \rho_t \frac{\partial \ell(x_t, \theta(\mu_t))}{\partial \theta} \end{aligned} \quad (11)$$

where $E_z[s(z, x_t|\theta(\mu_t))]$ denotes the expected sufficient statistics, $E_z[s(z, x_t|\theta(\mu_t))] = \int s(z, x_t) p(z|x_t, \theta(\mu_t)) dz$.

He proved the convergence of the above iteration method by casting it as a second order stochastic gradient descent using the following equality,

$$\frac{\partial \ell(x, \theta)}{\partial \theta} = \frac{\partial \mu}{\partial \theta} \frac{\partial \ell(x, \theta(\mu))}{\partial \mu} = I(\mu)^{-1} \frac{\partial \ell(x, \theta(\mu))}{\partial \mu} \quad (12)$$

This equality is obtained by firstly applying the chain rule, followed by the equality shown in Equation (5). It shows that online EM is equivalent to a stochastic gradient descent with $I(\mu_t)^{-1}$ as coefficient matrices [9].

Sato noted that the third term of the equality in Equation (12) resembles a natural gradient (see Theorem 1), but he did not explore the connection. But the key insights of the above derivation, which were not noted by Sato, is that Equation (12) is also valid for other loss functions different from the marginal log-likelihood; and that the convergence of Equation (11) does not depend on the formulation of the EM as a “lower bound maximization” method [22].

3 STOCHASTIC DISCRIMINATIVE EM

3.1 THE sdEM ALGORITHM

We consider the following supervised learning setup. Let us assume that we are given a data set D with n observations $\{(y_1, x_1), \dots, (y_n, x_n)\}$. We are also given a *discriminative loss function*² $\ell(y_i, x_i, \theta)$. For example, it could be the negative conditional log-likelihood (NCLL) $\ell(y_i, x_i, \theta) = -\ln p(y_i, x_i|\theta) + \ln \int p(y, x_i|\theta) dy = -\ln p(y_i|x_i, \theta)$. Our learning problem consists in minimizing the following objective function:

$$\begin{aligned} L(\theta) &= \sum_{i=1}^n \ell(y_i, x_i, \theta) - \ln p(\theta|\alpha) \\ &= E[\ell(y, x, \theta)|\pi] - \frac{1}{n} \ln p(\theta|\alpha) \end{aligned} \quad (13)$$

where π is now the empirical distribution of D and $E[\ell(y, x, \theta)|\pi]$ the empirical risk. Although the above

²The loss function is assumed to satisfy the mild conditions given in [9]. E.g., it can be a non-smooth function, such as the Hinge Loss.

loss function is not standard in the machine learning literature, we note that when ℓ is the negative log-likelihood (NLL), we get the classic *maximum a posterior estimation*. This objective function can be seen as an extension of this framework.

sdEM is presented as a generalization of Sato's online EM algorithm for finding the minimum of an objective function in the form of Equation (13) (i.e. the solution to our learning problem). The stochastic updating equation of sdEM can be expressed as follows,

$$\mu_{t+1} = \mu_t - \rho_t I(\mu_t)^{-1} \frac{\partial \bar{\ell}(y_t, x_t, \theta(\mu_t))}{\partial \mu} \quad (14)$$

where (y_t, x_t) denotes the t -th sample, randomly generated from π , and the function $\bar{\ell}$ has the following expression: $\bar{\ell}(y_t, x_t, \theta(\mu_t)) = \ell(y_t, x_t, \theta(\mu_t)) + 1/n \ln p(\theta(\mu_t))$. We note that this loss function satisfies the following equality, which is the base for a stochastic approximation method [19], $E[\bar{\ell}(y_t, x_t, \theta(\mu)) | \pi] = L(\theta(\mu))$.

Similarly to Amari's natural gradient algorithm [3], the main problem of sdEM formulated as in Equation (14) is the computation of the inverse of the Fisher information matrix at each step, which becomes even prohibitive for large models. The following result shows that this can be circumvented when we deal with distributions of the natural exponential family:

Theorem 2. *In a natural exponential family, the natural gradient of a loss function with respect to the expectation parameters equals the gradient of the loss function with respect to the natural parameters,*

$$I(\mu)^{-1} \frac{\partial \bar{\ell}(y, x, \theta(\mu))}{\partial \mu} = \frac{\partial \bar{\ell}(y, x, \theta)}{\partial \theta}$$

Sketch of the proof. We firstly need to prove that $I(\mu)$ is a valid Riemannian tensor metric and, hence, the expectation parameter space has a Riemannian structure defined by the metric $I(\mu)$ and the definition of the natural gradient makes sense. This can be proved by the invariant property of the Fisher information metric to one-to-one reparameterizations or, equivalently, transformations in the system of coordinates [2, 4]. $I(\mu)$ is a Riemannian metric because it is the Fisher information matrix of the reparameterized model $p(y, x | \theta(\mu))$, and the reparameterization is one-to-one, as commented in Section 2.2.

The equality stated in the theorem follows directly from Sato's derivation of the online EM algorithm (Equation (12)). This derivation shows that we can avoid the computation of $I(\mu)^{-1}$ by using the natural parameters instead of the expectation parameters and the function $\theta(\mu)$. \square

Theorem 1 simplifies the sdEM's updating equation to,

$$\mu_{t+1} = \mu_t - \rho_t \frac{\partial \bar{\ell}(y_t, x_t, \theta(\mu_t))}{\partial \theta} \quad (15)$$

sdEM can be interpreted as a stochastic gradient descent algorithm iterating over the *expectation parameters* and guided by the natural gradient in this Riemannian space.

Algorithm 1 Stochastic Discriminative EM (sdEM)

Require: D is randomly shuffled.

```

1:  $\mu_0 = \bar{\alpha}$ ; (initialize according to the prior)
2:  $\theta_0 = \theta(\mu_0)$ ;
3:  $t = 0$ ;
4: repeat
5:   for  $i = 1, \dots, n$  do
6:     E-Step:  $\mu_{t+1} = \mu_t - \frac{1}{(1+\lambda t)} \frac{\partial \bar{\ell}(y_i, x_i, \theta_t)}{\partial \theta}$ ;
7:     Check-Step:  $\mu_{t+1} = \text{Check}(\mu_{t+1}, \mathcal{S})$ ;
8:     M-Step:  $\theta_{t+1} = \theta(\mu_{t+1})$ ;
9:      $t = t + 1$ ;
10:   end for
11: until convergence
12: return  $\theta(\mu_t)$ ;
```

An alternative proof to Theorem 2 based on more recent results on information geometry has been recently given in [25]. The results of that work indicate that sdEM could also be interpreted as a mirror descent algorithm with a Bregman divergence as a proximity measure. It is beyond the scope of the paper to explore this relevant connection.

3.2 CONVERGENCE OF sdEM

In this section we do not attempt to give a formal proof of the convergence of sdEM, since very careful technical arguments would be needed for this purpose [9]. We simply go through the main elements that define the convergence of sdEM as an stochastic approximation method [19].

According to Equation (14), sdEM can be seen as a stochastic gradient descent method with the inverse of the Fisher information matrix $I(\mu)^{-1}$ as a coefficient matrix [9]. As we are dealing with natural exponential families, these matrices are always positive-definite. Moreover, if the gradient $\partial \bar{\ell}(y, x, \theta) / \partial \theta$ can be computed exactly (in Section 3.4 we discuss what happens when this is not possible), from Theorem 2, we have that it is an unbiased estimator of the natural gradient of the $L(\theta(\mu))$ defined in Equation 13,

$$E \left[\frac{\partial \bar{\ell}(y, x, \theta)}{\partial \theta} | \pi \right] = I(\mu)^{-1} \frac{\partial L(\theta(\mu))}{\partial \mu} \quad (16)$$

However, one key difference in terms of convergence between online EM and sdEM can be seen in Equation (11): μ_{t+1} is a convex combination between μ_t and the expected sufficient statistics. Then, $\mu_{t+1} \in \mathcal{S}$ during all the iterations. As will be clear in the next section, we do not have this same guarantee in sdEM, but we can take advantage

Table 1: sdEM updating equations for fully observed data (Section 3.3) .

Loss	sdEM equation
NLL	$\mu_{t+1} = (1 - \rho_t(1 + \frac{\nu}{n}))\mu_t + \rho_t (s(y_t, x_t) + \frac{1}{n}\bar{\alpha})$
NCLL	$\mu_{t+1} = (1 - \rho_t \frac{\nu}{n})\mu_t + \rho_t (s(y_t, x_t) - E_y[s(y, x_t) \theta(\mu_t)] + \frac{1}{n}\bar{\alpha})$
Hinge	$\mu_{t+1} = (1 - \rho_t \frac{\nu}{n})\mu_t + \rho_t \begin{cases} \frac{1}{n}\bar{\alpha} & \text{if } \ln \frac{p(y_t, x_t \theta)}{p(\bar{y}_t, x_t \theta)} > 1 \\ s(y_t, x_t) - s(\bar{y}_t, x_t) + \frac{1}{n}\bar{\alpha} & \text{otherwise} \end{cases}$ <p style="text-align: center;">where $\bar{y}_t = \arg \max_{y \neq y_t} p(y, x_t \theta)$</p>

of the log prior term of Equation (13) to avoid this problem. This term plays a dual role as both “regularization” term and log-barrier function [30] i.e. a continuous function whose value increases to infinity as the parameter approaches the *boundary of the feasible region* or the support of $p(\theta(\mu)|\alpha)$ ³. Then, if the step sizes ρ_t are small enough (as happens near convergence), sdEM will always stay in the feasible region \mathcal{S} , due to the effect of the log prior term. The only problem is that, in the initial iterations, the step sizes ρ_t are large, so one iteration can jump out of the boundary of \mathcal{S} . The method to avoid that depends on the particular model, but for the models examined in this work it seems to be a simple check in every iteration. For example, as we will see in the experimental section when implementing a multinomial Naive Bayes, we will check at every iteration that each sufficient statistic or “word count” is always positive. If a “word count” is negative at some point, we will set it to a very small value. As mentioned above, this does not hurt the convergence of sdEM because in the limit this problem disappears due the effect of the log-prior term.

The last ingredient required to assess the convergence of a stochastic gradient descent method is to verify that the sequence of step sizes satisfies: $\sum \rho_t = \infty$, $\sum \rho_t^2 < \infty$.

So, if the sequence $(\mu_t)_{t \geq 0}$ converges, it will probably converge to the global minimum $(\mu^*, \theta^* = \theta(\mu^*))$ if $L(\theta)$ is convex, or to a local minimum if $L(\theta)$ is not convex [9].

Finally, we give an algorithmic description of sdEM in Algorithm 1. Following [11], we consider steps sizes of the form $\rho_t = (1 + \lambda t)^{-1}$, where λ is a positive scalar⁴. As mentioned above, the “Check-Step” is introduced to guarantee that μ_t is always in \mathcal{S} . Like the online EM algorithm [29, 13], Algorithm 1 resembles the classic *expecta-*

tion maximization algorithm [15] since, as we will see in the next section, the gradient is computed using *expected sufficient statistics*. Assumption 3 guarantees that the maximization step can be performed efficiently. This step differentiates sdEM from classic stochastic gradient descent methods, where such a computation does not exist.

3.3 DISCRIMINATIVE LOSS FUNCTIONS

As we have seen so far, the derivation of sdEM is complete except for the definition of the loss function. We will discuss now how two well known *discriminative loss functions* can be used with this algorithm.

Negative Conditional Log-likelihood (NCLL)

As mentioned above, this loss function is defined as follows:

$$\ell_{CL}(y_t, x_t, \theta) = -\ln p(y_t, x_t|\theta) + \ln \int p(y, x_t|\theta) dy$$

And its gradient is computed as

$$\frac{\partial \ell_{CL}(y_t, x_t, \theta)}{\partial \theta} = -s(y_t, x_t) + E_y[s(y, x_t)|\theta]$$

where the sufficient statistic $s(y_t, x_t)$ comes from the gradient of the $\ln p(y_t, x_t|\theta)$ term in the NCLL loss, and the expected sufficient statistic $E_y[s(y, x_t)|\theta] = \int s(y, x_t)p(y|x_t, \theta)dy$, comes from the gradient of the $\ln \int p(y, x_t|\theta)dy$ term in the NCLL loss. As mentioned above, the computation of the gradient is similar to the *expectation step* of the classic EM algorithm.

The iteration equation of sdEM for the NCLL loss is detailed in Table 1. We note that in the case of multi-class prediction problems the integrals of the updating equation are replaced by sums over the different classes of the class variable Y . We also show the updating equation for the negative log-likelihood (NLL) loss for comparison purposes.

³The prior p would need to be suitably chosen.

⁴Our experiments suggest that trying $\lambda \in \{1, 0.1, 0.01, 0.001, \dots\}$ suffices for obtaining a quick convergence.

Table 2: sdEM updating equations for partially observed data (Section 3.4)

Loss	sdEM equation
NLL	$\mu_{t+1} = (1 - \rho_t(1 + \frac{\nu}{n}))\mu_t + \rho_t (E_z[s(y_t, z, x_t) \theta(\mu_t)] + \frac{1}{n}\bar{\alpha})$
NCLL	$\mu_{t+1} = (1 - \rho_t\frac{\nu}{n})\mu_t + \rho_t (E_z[s(y_t, z, x_t) \theta(\mu_t)] - E_{y_z}[s(y, z, x_t) \theta(\mu_t)] + \frac{1}{n}\bar{\alpha})$
Hinge	$\mu_{t+1} = (1 - \rho_t\frac{\nu}{n})\mu_t + \rho_t \begin{cases} \frac{1}{n}\bar{\alpha} & \text{if } \ln \frac{\int p(y_t, z, x_t \theta) dz}{\int p(\bar{y}_t, z, x_t \theta) dz} > 1 \\ E_z[s(y_t, z, x_t) \theta(\mu_t)] - E_z[s(\bar{y}_t, z, x_t) \theta(\mu_t)] + \frac{1}{n}\bar{\alpha} & \text{otherwise} \end{cases}$ <p style="text-align: right;">where $\bar{y}_t = \arg \max_{y \neq y_t} \int p(y, z, x_t \theta) dz$</p>

The Hinge loss

Unlike the previous loss which is valid for continuous and discrete (and vector-valued) predictions, this loss is only valid for binary or multi-class classification problems.

Margin-based loss functions have been extensively used and studied by the machine learning community for binary and multi-class classification problems [5]. However, in our view, the application of margin-based losses (different from the negative conditional log-likelihood) for discriminative training of probabilistic generative models is scarce and based on ad-hoc learning methods which, in general, are quite sophisticated [24]. In this section, we discuss how sdEM can be used to minimize the empirical risk of one of the most used margin-based losses, the Hinge loss, in binary and multi-class classification problems. But, firstly, we discuss how Hinge loss can be defined for probabilistic generative models.

We build on LeCun et al.'s ideas [21] about energy-based learning for prediction problems. LeCun et al. [21] define the Hinge loss for energy-based models as follows,

$$\max(0, 1 - (E(\bar{y}_t, x_t, w) - E(y_t, x_t, w)))$$

where $E(\cdot)$ is the energy function parameterized by a parameter vector w , $E(y_t, x_t, w)$ is the energy associated to the correct answer y_t and $E(\bar{y}_t, x_t, w)$ is the energy associated to the most offending incorrect answer, $\bar{y}_t = \arg \min_{y \neq y_t} E(y, x_t, w)$. Predictions y^* are made using $y^* = \arg \min_y E(y, x_t, w^*)$ when the parameter w^* that minimizes the empirical risk is found.

In our learning settings we consider the minus logarithm of the joint probability, $-\ln p(y_t, x_t|\theta)$, as an energy function. In consequence, we define the hinge loss as follows

$$\ell_{\text{hinge}}(y_t, x_t, \theta) = \max(0, 1 - \ln \frac{p(y_t, x_t|\theta)}{p(\bar{y}_t, x_t|\theta)}) \quad (17)$$

where \bar{y}_t denotes here too the most offending incorrect answer, $\bar{y}_t = \arg \max_{y \neq y_t} p(y, x_t|\theta)$.

The gradient of this loss function can be simply computed as follows

$$\frac{\partial \ell_{\text{hinge}}(y_t, x_t, \theta)}{\partial \theta} = \begin{cases} 0 & \text{if } \ln \frac{p(y_t, x_t|\theta)}{p(\bar{y}_t, x_t|\theta)} > 1 \\ -s(y_t, x_t) + s(\bar{y}_t, x_t) & \text{otherwise} \end{cases}$$

and the iteration equation for minimizing the empirical risk of the Hinge loss is also given in Table 1.

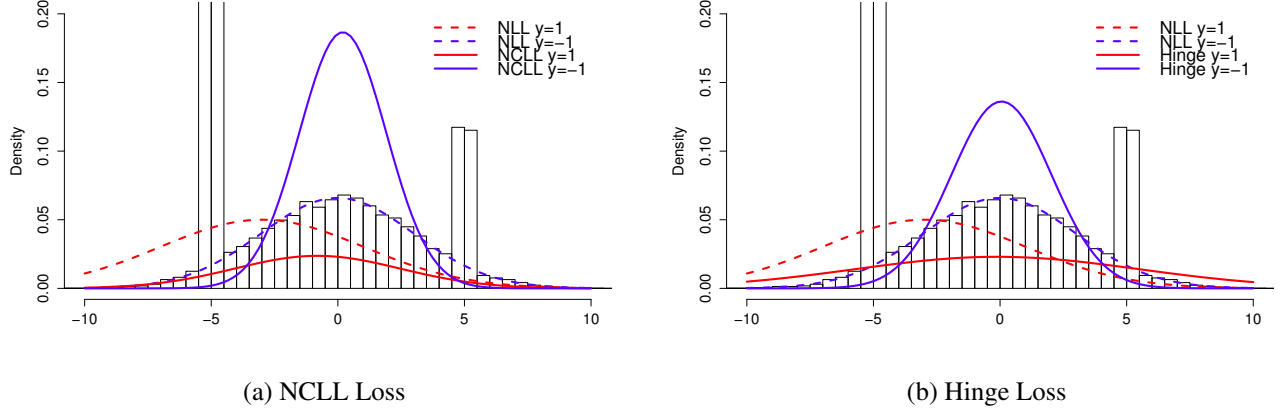
3.4 PARTIALLY OBSERVABLE DATA

The generalization of sdEM to partially observable data is straightforward. We denote by Z the vector of non-observable variables. sdEM will handle statistical models which define a probability distribution over (y, z, x) which belongs to the natural exponential family (Assumption 1). Assumption 2 and 3 remain unaltered.

The tuple (y, z, x) will denote the complete event or complete data, while the tuple (y, x) is the observed event or the observed data. So we assume that our given data set D with n observations is expressed as $\{(y_1, x_1), \dots, (y_n, x_n)\}$. So sdEM's Equation (14) and (15) are the same, with the only difference that the natural gradient is now defined using the inverse of the Fisher information matrix for the statistical model $p(y, z, x|\theta(\mu))$. The same happens for Theorem 2.

The NCLL loss and the Hinge loss are equally defined as in Section 3.3, with the only difference that the computation of $p(y_t, x_t|\theta)$ and $p(x_t|\theta)$ requires marginalization over z , $p(y_t, x_t|\theta) = \int p(y_t, z, x_t|\theta) dz$, $p(x_t|\theta) = \int p(y, z, x_t|\theta) dy dz$. The updating equations for sdEM under partially observed data for the NCLL and Hinge loss are detailed in Table 2. New expected sufficient statistics need to be computed,

Figure 1: Toy example (Section 4.1). The result using the NLL loss (i.e. MLE estimation) is plotted with dashed lines which represent the densities $p(y = k)N(x, \mu^{(k)}, \sigma^{(k)})$ for both classes (i.e. when the red line is higher than the blue line we predict the red class and vice versa). The estimated prediction accuracy of the MLE model is 78.6%. Solid lines represent the same estimation but using the NCLL and the Hinge loss. Their estimated prediction accuracies are 90.4% and 90.6%, respectively.



$E_z[s(y_t, z, x_t)|\theta] = \int s(y_t, z, x_t)p(z|y_t, x_t, \theta)dz$ and $E_{yz}[s(y, z, x_t)|\theta] = \int s(y, z, x_t)p(y, z|x_t, \theta)dydz$. As previously, we also show the updating equation for the negative log-likelihood (NLL) loss for comparison purposes.

3.5 sdEM AND APPROXIMATE INFERENCE

For many interesting models [8], the computation of the expected sufficient statistics in the iteration equations shown in Table 1 and 2 cannot be computed in closed form. This is not a problem as far as we can define *unbiased estimators* for these expected sufficient statistics, since the equality of Equation (16) still holds. As it will be shown in the next section, we use sdEM to discriminatively train *latent Dirichlet allocation* (LDA) models [8]. Similarly to [26], for this purpose we employ collapsed Gibbs sampling to compute the expected sufficient statistics, $E_z[s(y_t, z, x_t)|\theta]$, as it guarantees that at convergence samples are i.i.d. according to $p(z|y_t, x_t, \theta)$.

4 EXPERIMENTAL ANALYSIS

4.1 TOY EXAMPLE

We begin the experimental analysis of sdEM by learning a very simple Gaussian naive Bayes model composed by a binary class variable Y and a single continuous predictor X . Hence, the conditional density of the predictor given the class variable is assumed to be normally distributed. The interesting part of this toy example is that the training data is generated by a different model: $\pi(y = -1) = 0.5$, $\pi(x|y = -1) \sim N(0, 3)$ and $\pi(x|y = 1) \sim$

$0.8 \cdot N(-5, 0.1) + 0.2 \cdot N(5, 0.1)$. Figure 1 shows the histogram of the 30,000 samples generated from the π distribution. The result is a mixture of 3 Gaussians, one in the center with a high variance associated to $y = -1$ and two narrow Gaussians on both sides associated to $y = 1$.

sdEM can be used by considering 6 (non-minimal) sufficient statistics: $N^{(-1)}$ and $N^{(1)}$ as “counts” associated to both classes, respectively; $S^{(-1)}$ and $S^{(1)}$ as the “sum” of the x values associated to classes $y = -1$ and $y = 1$, respectively; and $V^{(-1)}$ and $V^{(1)}$ as the “sum of squares” of the x values for each class. We also have five parameters which are computed from the sufficient statistics as follows: Two for the prior of class $p(y = -1) = p^{(-1)} = N^{(-1)} / (N^{(-1)} + N^{(1)})$ and $p^{(1)} = N^{(1)} / (N^{(-1)} + N^{(1)})$; and four for the two Gaussians which define the conditional of X given Y , $\mu^{(-1)} = S^{(-1)} / N^{(-1)}$, $\sigma^{(-1)} = \sqrt{V^{(-1)} / N^{(-1)} - (S^{(-1)} / N^{(-1)})^2}$, and equally for $\mu^{(1)}$ and $\sigma^{(1)}$.

The sdEM’s updating equations for the NCLL loss can be written as follows

$$\begin{aligned} N_{t+1}^{(k)} &= N_t^{(k)} + \rho_t (I[y_t = k] - p_t(k|x_t)) + \frac{\rho_t}{n} \\ S_{t+1}^{(k)} &= (1 - \frac{\rho_t}{n}) S_t^{(k)} + \rho_t x_t (I[y_t = k] - p_t(k|x_t)) \\ V_{t+1}^{(k)} &= (1 - \frac{\rho_t}{n}) V_t^{(k)} + \rho_t x_t^2 (I[y_t = k] - p_t(k|x_t)) + \frac{\rho_t}{n} \end{aligned}$$

where k indexes both classes, $k \in \{-1, 1\}$, $I[\cdot]$ denotes the indicator function, $p_t(k|x_t)$ is an abbreviation of $p(y = k|x_t, \theta_t)$, and θ_t is the parameter vector computed from the sufficient statistics at the t -th iteration.

Figure 2: Convergence trade-off of the Hinge loss versus the NCLL loss and the perplexity for a multinomial naive Bayes model trained minimizing the Hinge loss using sdEM. Circle-lines, triangle-lines and cross-lines correspond to the results with 20NewsGroup, Cade and Reuters-R52 datasets, respectively.

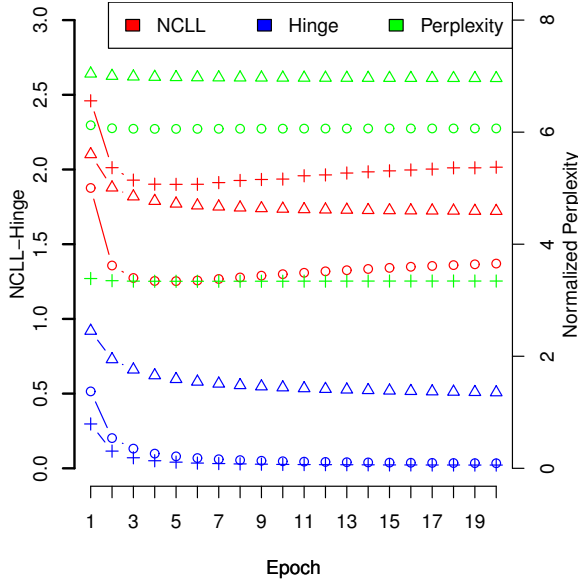
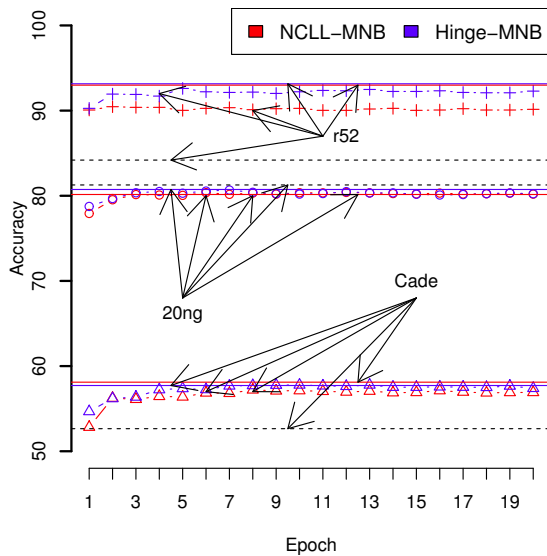


Figure 3: Convergence of the classification accuracy for a multinomial naive Bayes model trained minimizing the NCLL loss (NCLL-MNB) and the Hinge loss (Hinge-MNB) using sdEM. Red circle-lines, red triangle-lines and red cross-lines correspond to the results of NCLL-MNB with 20NewsGroup, Cade and Reuters-R52 datasets, respectively. Same for Hinge-MNB. The three blue and the three red solid lines detail the accuracy of logistic regression and SVM, respectively. The three dashed black lines detail the accuracy of plain MNB with a Laplace prior.



Similarly, the sdEM's updating equations for the Hinge loss can be written as follows,

$$N_{t+1}^{(k)} = N_t^{(k)} + ky_t \rho_t I[\ln \frac{p_t(y_t|x_t)}{p_t(\bar{y}_t|x_t)} < 1] + \frac{\rho_t}{n}$$

$$S_{t+1}^{(k)} = (1 - \frac{\rho_t}{n})S_t^{(k)} + ky_t \rho_t x_t I[\ln \frac{p_t(y_t|x_t)}{p_t(\bar{y}_t|x_t)} < 1]$$

$$V_{t+1}^{(k)} = (1 - \frac{\rho_t}{n})V_t^{(k)} + ky_t \rho_t x_t^2 I[\ln \frac{p_t(y_t|x_t)}{p_t(\bar{y}_t|x_t)} < 1] + \frac{\rho_t}{n}$$

where the product ky_t is introduced in the updating equations to define the sign of the sum, and the indicator function $I[\cdot]$ defines when the hinge loss is null.

In the above set of equations we have considered as a conjugate prior for the Gaussians a three parameter Normal-Gamma prior, $\nu = 1$ and $\bar{\alpha}_1 = 0$ for $S^{(k)}$ and $\bar{\alpha}_2 = 1$ for $V^{(k)}$ [6, page 268], and a Beta prior with $\nu = 0$ and $\bar{\alpha} = 1$ for $N^{(k)}$. We note that these priors assign zero probability to “extreme” parameters $p^{(k)} = 0$ (i.e. $N^{(k)} = 0$) and $\sigma^{(k)} = 0$ (i.e. $V^{(k)}/N^{(k)} - (S^{(k)}/N^{(k)})^2 = 0$).

Finally, the “Check-step” (see Algorithm 1) performed before computing θ_{t+1} , and which guarantees that all sufficient statistics are correct, is implemented as follows:

$$N_{t+1}^{(k)} = \max(N_{t+1}^{(k)}, \frac{\rho_t}{n})$$

$$V_{t+1}^{(k)} = \max(V_{t+1}^{(k)}, \frac{(S_{t+1}^{(k)})^2}{N_{t+1}^{(k)}} + \frac{\rho_t}{n})$$

I.e., when the $N^{(k)}$ “counts” are negative or too small or when the $V^{(k)}$ values lead to negative or null deviations $\sigma^{(k)} \leq 0$, they are fixed with the help of the prior term.

The result of this experiment is given in Figure 1 and clearly shows the different trade-offs of both loss functions compared to *maximum likelihood estimation*. It is interesting to see how a generative model which does not match the underlying distribution is able to achieve a pretty high prediction accuracy when trained with a discriminative loss function (using the sdEM algorithm).

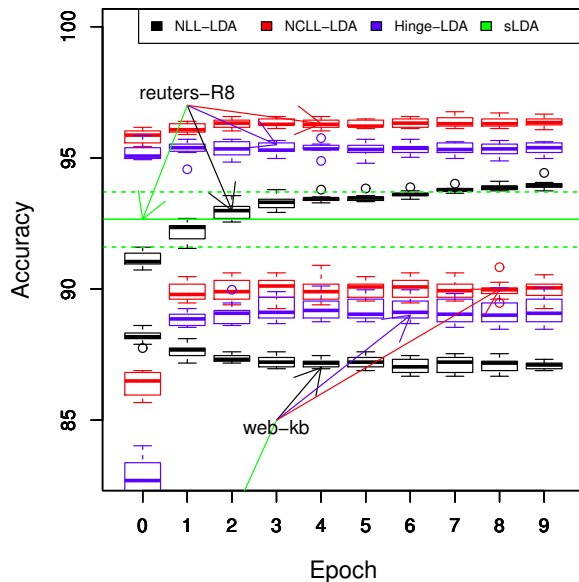
4.2 sdEM FOR TEXT CLASSIFICATION

Next, we briefly show how sdEM can be used to discriminatively train some generative models used for text classification, such as multinomial naive Bayes and a similar classifier based on latent Dirichlet allocation models [8]. Supplementary material with full details of these experiments and the Java code used in this evaluation can be download at: <http://sourceforge.net/projects/sdem/>

Multinomial Naive Bayes (MNB)

MNB assumes that words in documents with the same class or label are distributed according to an independent multinomial distribution. sdEM can be easily applied to train this

Figure 4: Convergence of the classification accuracy of LDA classification models trained by sdEM using different loss functions (NLL, NCLL and Hinge) over 10 different random initializations. The two dashed lines and the single solid line detail the maximum, minimum and mean accuracy of sLDA, respectively, over 10 random initializations.



model. The sufficient statistics are the “prior class counts” and the “word counts” for each class. The updating equations and the check step are the same as those of $N_t^{(k)}$ in the previous toy example. Parameters of the MNB are computed simply through normalization operations. Two different conjugate Dirichlet distributions were considered: A “Laplace prior” where $\bar{\alpha}_i = 1$; and a “Log prior” where $\bar{\alpha}_i = \text{“logarithm of the number of words in the corpus”}$. We only report analysis for “Laplace prior” in the case of NCLL loss and for “Log prior” in the case of Hinge loss. Other combinations show similar results, although NCLL was more sensitive to the chosen prior.

We evaluate the application of sdEM to MNB with three well-known multi-class text classification problems: 20Newsgroup (20 classes), Cade (12 classes) and Reuters21578-R52 (52 classes). Data sets are stemmed. Full details about the data sets and the train/test data sets split used in this evaluation can be found in [14].

Figure 2 shows the convergence behavior of sdEM with $\lambda = 1e-05$ when training a MNB by minimizing the Hinge loss (Hinge-MNB). In this figure, we plot the evolution of the Hinge loss but also the evolution of the NCLL loss and the normalized perplexity (i.e. the perplexity measure [8] divided by the number of training documents) at each epoch. We can see that there is a trade-off between the different losses. E.g., Hinge-MNB decreases the Hinge loss (as expected) but tends to increase the NCLL loss, while it

only decreases perplexity at the very beginning.

Figure 3 displays the evolution of the classification accuracy of two MNBs trained minimizing the NCLL loss and the Hinge loss using sdEM. We compare them to: the standard MNB with a “Laplace prior”; the L2-regularized Logistic Regression; and the primal L2-regularized SVM. The two later methods were taken from the Liblinear toolkit v.18 [17]. As can be seen, sdEM is able to train simple MNB models with a performance very close to that provided by highly optimized algorithms.

Latent Dirichlet Allocation (LDA)

We briefly show the results of sdEM when discriminatively training LDA models. We define a classification model equal to MNB, but where the documents of the same class are now modeled using an independent LDA model. We implement this model by using, apart from the “prior class counts”, the standard sufficient statistics of the LDA model, i.e. “words per hidden topic counts”, associated to each class label. Similarly to [26], we used an online Collapsed Gibbs sampling method to obtain, at convergence, unbiased estimates of the expected sufficient statistics (see Table 2).

This evaluation was carried out using the standard train/test split of the Reuters21578-R8 (8 classes) and web-kb (4 classes) data sets [14], under the same preprocessing than in the MNB’s experiments. Figure 4 shows the results of this comparison using 2-topics LDA models trained with the NCLL loss (NCLL-LDA), the Hinge loss (Hinge-LDA), and also the NLL loss (NLL-LDA) following the updating equations of Table 2. We compared these results with those returned by supervised-LDA (sLDA) [7] using the same prior, but this time with 50 topics because less topics produced worse results. We see again how a simple generative model trained with sdEM outperforms much more sophisticated models.

5 CONCLUSIONS

We introduce a new learning algorithm for discriminative training of generative models. This method is based on a novel view of the online EM algorithm as a stochastic *natural gradient* descent algorithm for minimizing general discriminative loss functions. It allows the training of a wide set of generative models with or without latent variables, because the resulting models are always generative. Moreover, sdEM is comparatively simpler and easier to implement (and debug) than other ad-hoc approaches.

Acknowledgments

This work has been partially funded from the European Union’s Seventh Framework Programme for research, technological development and demonstration under grant agreement no 619209 (AMIDST project).

References

- [1] Sungjin Ahn, Anoop Korattikara Balan, and Max Welling. Bayesian posterior sampling via stochastic gradient Fisher scoring. In *ICML*, 2012.
- [2] Shun-ichi Amari. *Differential-geometrical methods in statistics*. Springer, 1985.
- [3] Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural Comput.*, 10(2):251–276, 1998.
- [4] Shun-ichi Amari and Hiroshi Nagaoka. *Methods of information geometry*, volume 191. American Mathematical Soc., 2007.
- [5] Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.
- [6] José M Bernardo and Adrian FM Smith. *Bayesian theory*, volume 405. John Wiley & Sons, 2009.
- [7] David M Blei and Jon D McAuliffe. Supervised topic models. In *NIPS*, volume 7, pages 121–128, 2007.
- [8] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.
- [9] L. Bottou. Online learning and stochastic approximations. *On-line learning in neural networks*, 17(9), 1998.
- [10] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT’2010*, pages 177–186. Springer, 2010.
- [11] L. Bottou. Stochastic gradient descent tricks. In *Neural Networks: Tricks of the Trade*, pages 421–436. Springer, 2012.
- [12] L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In *NIPS*, volume 4, page 2, 2007.
- [13] Olivier C. and E. Moulines. On-line expectation-maximization algorithm for latent data models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(3):593–613, 2009.
- [14] Ana Cardoso-Cachopo. Improving Methods for Single-label Text Categorization. PdD Thesis, 2007.
- [15] Arthur P Dempster, Nan M Laird, Donald B Rubin, et al. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal statistical Society*, 39(1):1–38, 1977.
- [16] Greiner et al. Structural extension to logistic regression: Discriminative parameter learning of belief net classifiers. *Mach. Learning*, 59(3):297–322, 2005.
- [17] R.E. Fan, K.W. Chang, C.J. Hsieh, X.R. Wang, and C.J. Lin. Liblinear: A library for large linear classification. *JMLR*, 9:1871–1874, 2008.
- [18] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.
- [19] Harold Joseph Kushner and G George Yin. *Stochastic approximation algorithms and applications*. Springer New York, 1997.
- [20] S. Lacoste-Julien, F. Sha, and M.I. Jordan. DiscLDA: Discriminative learning for dimensionality reduction and classification. In *NIPS*, volume 83, page 85, 2008.
- [21] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. *Predicting structured data*, 2006.
- [22] Radford M Neal and Geoffrey E Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pages 355–368. Springer, 1998.
- [23] Frank Nielsen and Vincent Garcia. Statistical exponential families: A digest with flash cards. *arXiv preprint arXiv:0911.4863*, 2009.
- [24] F. Pernkopf, M. Wohlmayr, and S. Tschitschek. Maximum margin Bayesian network classifiers. *IEEE Trans. PAMI*, 34(3):521–532, 2012.
- [25] Garvesh Raskutti and Sayan Mukherjee. The information geometry of mirror descent. *arXiv preprint arXiv:1310.7780*, 2013.
- [26] D. Rohde and O. Cappe. Online maximum-likelihood estimation for latent factor models. In *Statistical Signal Processing Workshop (SSP), 2011 IEEE*, pages 565–568, June 2011.
- [27] David Ruppert. Efficient estimations from a slowly convergent Robbins-Monro process. Technical report, Cornell University Operations Research and Industrial Engineering, 1988.
- [28] Masa-aki Sato. Fast learning of on-line EM algorithm. *Rapport Technique, ATR Human Information Processing Research Laboratories*, 1999.
- [29] Masa-aki Sato. Convergence of on-line EM algorithm. In *Proc. of the Int. Conf. on Neural Information Processing*, volume 1, pages 476–481, 2000.
- [30] SJ Wright and J Nocedal. *Numerical optimization*, volume 2. Springer New York, 1999.
- [31] Jun Zhu, Amr Ahmed, and Eric P Xing. MedLDA: maximum margin supervised topic models for regression and classification. In *ICML*, pages 1257–1264. ACM, 2009.

Latent Kullback Leibler Control for Continuous-State Systems using Probabilistic Graphical Models

Takamitsu Matsubara[†]

Vicenç Gómez^{§,*}

Hilbert J. Kappen[§]

[†] Graduate School of Information Science. Nara Institute of Science and Technology (NAIST). Nara, Japan

[§] Donders Institute for Brain, Cognition and Behaviour. Radboud University Nijmegen, the Netherlands

^{*} Department of Information and Communication Technologies. Universitat Pompeu Fabra. Barcelona, Spain

Abstract

Kullback Leibler (KL) control problems allow for efficient computation of optimal control by solving a principal eigenvector problem. However, direct applicability of such framework to continuous state-action systems is limited. In this paper, we propose to embed a KL control problem in a probabilistic graphical model where observed variables correspond to the continuous (possibly high-dimensional) state of the system and latent variables correspond to a discrete (low-dimensional) representation of the state amenable for KL control computation. We present two examples of this approach. The first one uses standard hidden Markov models (HMMs) and computes exact optimal control, but is only applicable to low-dimensional systems. The second one uses factorial HMMs, it is scalable to higher dimensional problems, but control computation is approximate. We illustrate both examples in several robot motor control tasks.

1 INTRODUCTION

Recent research in stochastic optimal control theory has identified a class of problems known as Kullback-Leibler (KL) control problems (Kappen et al., 2012) or linearly solvable Markov decision problems (LSMDPs) (Todorov, 2006). For these (discrete) problems, the set of actions and the cost function are restricted in a way that makes the Bellman equation linear and thus more efficiently solvable, for instance, by solving the principal eigenvector of a certain linear operator (Todorov, 2009a).

However, direct applicability of this framework to continuous state-action systems, such as robot motor control, is limited. The main problem is the curse of

dimensionality, which appears because discretization quickly leads to a combinatorial explosion. This problem has been addressed using function approximation methods in (Todorov, 2009b). Instead of directly solving a discrete-state LSMDP, these methods approximate the so-called desirability function, which is defined in the continuous-state space. Kinjo et al. (2013) combined this function approximation scheme with system identification on a real robot navigation task. However, approaches based on the continuous-state formulation of KL control problems have several limitations: they require to solve a quadratic programming problem, a more computationally demanding problem than computing the principal eigenvector. Also, there is no guarantee of convergence to a positive solution. Alternative formulations that address these limitations have been recently proposed (Zhong and Todorov, 2011a,b). Zhong and Todorov (2011a) used a soft aggregation method to solve KL-control problems in an aggregated space. Both approaches, however, require the model of system dynamics, which is often not available in real-world applications (Kinjo et al., 2013).

In this paper, we propose to embed a KL-control problem in a probabilistic graphical model with mixed continuous and discrete variables. The continuous variables correspond to the (possibly high-dimensional) state of the system and the discrete variables correspond to a latent (low-dimensional) representation of the state which is amenable for KL control computation. The model parameters are first learned using data from the real system running with exploring controls. The control input to the real system is then computed as a filtering step combined with the solution of the KL-control problem in the latent space.

We present two examples of this approach: the first one uses a standard hidden Markov model (HMM) in which inference can be computed exactly, but is only applicable to low-dimensional continuous systems. The second one uses factorial HMMs (FHMMs)

and is applicable to higher dimensional problems, although optimal control can only be approximated. We illustrate both examples in several robot motor control tasks. In particular, we experimentally demonstrate that the second example with FHMMs is scalable to high-dimensional problems (e.g., 25 dimensional problem) that may not be solvable by other approaches.

2 KULLBACK LEIBLER CONTROL PROBLEMS

We briefly summarize the class of KL control problems introduced by Todorov (2006) in the infinite-horizon average-cost formulation (see also Todorov, 2009a).

Let $\mathcal{X} = \{1, \dots, N\}$ be a finite set of states and $\mathcal{U}(x)$ be a set of admissible control actions at state $x \in \mathcal{X}$. Consider the transition probability $p(x'|x)$ that describes the system dynamics in the absence of control. Such *uncontrolled dynamics* assigns zero probability for physically forbidden state transitions. Denote the transition probability given action $u \in \mathcal{U}(x)$ as $p(x'|x, u)$ and the immediate cost for being in state x and taking action u as $\ell(x, u) \geq 0$.

For infinite-horizon problems, the objective is to find a control law $u = \pi(x)$ that minimizes the average cost

$$\lim_{n \rightarrow \infty} \frac{1}{n} \mathbb{E} \left[\sum_{t=0}^{n-1} \ell(x_t, \pi(x_t)) \right] = \sum_x \Pi(x) \ell(x, \pi(x)) \quad (1)$$

where n is the number of time-steps and $\Pi(x) = \lim_{t \rightarrow \infty} p(x_t = x | x_0, \pi)$ is the stationary distribution of states under control law π , which we assume exists and is independent of x_0 , i.e., $p(x_t = x | x_0, \pi)$ is assumed ergodic.

The following Bellman equation defined for the (differential) cost-to-go function $v(x)$ minimizes Eq. (1)

$$c + v(x) = \min_{u \in \mathcal{U}(x)} \left\{ \ell(x, u) + \mathbb{E}_{x' \sim p(\cdot | x, u)} [v(x')] \right\}, \quad (2)$$

where c is the average cost that does not depend on the starting state.

Minimizing Eq. (2) is in general hard, but in some cases it can be done efficiently. KL control problems are a class of problems for which Eq. (2) becomes linear under the following assumptions:

(i) the controls directly specify state transition probabilities, i.e. $p(x'|x, u) = u(x'|x)$. The action vector $u(\cdot|x)$ is a probability distribution over next states given the current state x .

(ii) the immediate cost function has the following form

$$\ell(x, u) = \alpha q(x) + \text{KL}(u(\cdot|x) \parallel p(\cdot|x)),$$

where $q(x) \geq 0$ is an arbitrary state-dependent cost and KL is the Kullback Leibler divergence between the controlled and the uncontrolled dynamics, reflecting how much the control changes the normal behavior of the system. Parameter α allows to balance the two cost terms.

Define the exponentiated cost-to-go (desirability) function $z(x) = \exp(-v(x))$ and the linear operator

$$\mathcal{G}[z](x) = \sum_{x'} p(x'|x) z(x') = \mathbb{E}_{x' \sim p(\cdot | x, u)} [z(x')].$$

The resulting minimization takes the form

$$\min_{u \in \mathcal{U}(x)} \left\{ \alpha q(x) + \text{KL} \left(u(\cdot|x) \parallel \frac{p(\cdot|x)z(\cdot)}{\mathcal{G}[z](x)} \right) - \log \mathcal{G}[z](x) \right\}.$$

At the global minimum, the Bellman equation becomes

$$\exp(-c)z(x) = \exp(-\alpha q(x))\mathcal{G}[z](x)$$

or in matrix form

$$\lambda \mathbf{z} = \mathbf{G} \mathbf{P} \mathbf{z} \quad (3)$$

where \mathbf{G} is a $N \times N$ diagonal matrix with elements $\exp(-\alpha q(x))$ and $\lambda = \exp(-c)$. From Eq. (3), it follows that \mathbf{z} is any eigenvector of the matrix $\mathbf{G} \mathbf{P}$ with eigenvalue λ . The optimal average cost becomes $c = -\ln \lambda$. Thus, the minimal solution is given by the principal eigenvector of $\mathbf{G} \mathbf{P}$: the eigenvector \mathbf{z}^* with largest eigenvalue, which can be efficiently computed using the power iteration method (Todorov, 2006). The optimal control is given by

$$u^*(x'|x) = \frac{p(x'|x)\mathbf{z}^*(x')}{\mathcal{G}[\mathbf{z}^*](x)}. \quad (4)$$

3 LATENT KULLBACK LEIBLER CONTROL

The previously described framework is not directly applicable for continuous systems. For such cases, we propose to learn a discrete hidden representation and dynamics amenable for efficient computation from the observed continuous variables. Our approach can be summarized in the following three steps:

1. Learn a probabilistic graphical model from data samples obtained for the real system
2. Solve the KL control problem in the latent space of the probabilistic graphical model
3. Compute control in the observed space

This general method is directly applicable to arbitrary continuous state-action systems, while in this paper we focus on the following deterministic control-affine systems that typically describe discrete-time robot dynamics:

$$\mathbf{y}_{t+1} = \mathbf{y}_t + \Delta t (\mathbf{f}(\mathbf{y}_t) + \mathbf{B}(\mathbf{y}_t)\boldsymbol{\tau}_t), \quad (5)$$

where $\mathbf{y}_t \in \mathbb{R}^D$ is the state variable of the system, $\boldsymbol{\tau}_t \in \mathbb{R}^d$ is the control input, $\mathbf{f}(\mathbf{y}_t) \in \mathbb{R}^D$ is the uncontrolled dynamics, $\mathbf{B}(\mathbf{y}_t) \in \mathbb{R}^{D \times d}$ is the control matrix and Δt is the discrete-time step-size.

Two particular realizations of this general approach are described in the next section. The first one uses standard HMMs, which are the most natural way to model sequences of observations. However, it is only applicable to systems in which the relevant region of the state-space is small, such as low-dimensional systems, or largely constrained high-dimensional systems. The second one uses factorial HMMs, which assume factorized uncontrolled dynamics and can scale up to higher dimensional problems.

4 EXACT CONTROL COMPUTATION USING HIDDEN MARKOV MODELS

In this section, we describe an example of latent KL control based on standard hidden Markov models.

4.1 LEARNING HMMS FOR KL CONTROL

Consider the hidden Markov model with hidden states $x_t \in \{1, \dots, N\}$, stochastic state transition matrix \mathbf{P} with entries $P_{ij} = p(x_{t+1} = j | x_t = i)$ and Gaussian observation model $p(\mathbf{y}_t | x_t = k) = \mathcal{N}(\mu_k, \Sigma_k)$.

We generate sample trajectories $\mathcal{D} = \{\mathbf{y}_t, \dots, \mathbf{y}_T\}$ from the real system driven solely by exploration noise (uncontrolled dynamics) and use them to learn the parameters $\boldsymbol{\theta}_{\text{HMM}} = \{\mathbf{P}, \mu_{1:N}, \Sigma_{1:N}\}$. After learning, the matrix \mathbf{P} encodes a coarse description of the observed dynamics in a latent space and the Gaussian means and variances capture the relevant regions in this space. More precisely, considering the system of Eq. (5), we set exploration noise as $\boldsymbol{\tau}_t = \epsilon_t$ for $t = 1 \dots T$, where $\epsilon_t \in \mathbb{R}^d \sim \mathcal{N}(0, \Sigma_\epsilon)$. The choice of such a zero-mean Gaussian distribution is motivated by the relationship between the KL action cost and the input-norm cost: in the continuous setting the KL cost reduces to a quadratic energy cost (Todorov, 2009a; Kappen et al., 2012), which coincides with a commonly used input-norm cost for energy-efficient or smooth motor control behavior (Mitrovic et al., 2010).

The covariance matrix Σ_ϵ is a free parameter. For low exploration noise, one would expect the learned model to be a poor approximation since only a small fraction of the state space is visited. Conversely, large noise values would result in too flexible models with unrealistic state transitions. The correct noise value is therefore a trade-off between these two scenarios.

Given \mathcal{D} , the parameters $\boldsymbol{\theta}_{\text{HMM}}$ can be learned, for instance, using the standard Expectation-Maximization (EM) algorithm (Baum-Welch algorithm).

4.2 CONTROL COMPUTATION IN LATENT SPACE

To define a KL control problem in the latent space, we first need a state-dependent cost function expressed in terms of the latent variable x . Let $\tilde{q}(\mathbf{y}_t)$ and $q(x_t)$ be the cost functions in observation and latent spaces, respectively. We define $q(x_t)$ given $\tilde{q}(\mathbf{y}_t)$ using $\exp(-q(x_t)) = \int_{\mathbf{y}_t} \exp(-\alpha \tilde{q}(\mathbf{y}_t)) p(\mathbf{y}_t | x_t) d\mathbf{y}_t$. Furthermore, if $\tilde{q}(\mathbf{y}_t)$ is given in quadratic form $\tilde{q}(\mathbf{y}_t) = (\mathbf{y}_t - \mu_q)^T \Sigma_q^{-1} (\mathbf{y}_t - \mu_q) = \|\mathbf{y}_t - \mu_q\|_{\Sigma_q^{-1}}^2$ and the observation model is Gaussian $p(\mathbf{y}_t | x_t) = \mathcal{N}(\mu_x, \Sigma_x)$, we can obtain $q(x_t)$ analytically:

$$\begin{aligned} q(x_t) &= -\ln \left\{ \int_{\mathbf{y}_t} \exp(-\alpha \tilde{q}(\mathbf{y}_t)) p(\mathbf{y}_t | x_t) d\mathbf{y}_t \right\} \\ &= -\ln \left\{ \frac{|\mathbf{S}|^{1/2}}{|\Sigma_x|^{1/2}} \exp \left[-\frac{1}{2} \|\mu_q - \mu_x\|_{\mathbf{M}^{-1}}^2 \right] \right\} \end{aligned}$$

where, $\mathbf{S} = (\alpha \Sigma_q^{-1} + \Sigma_x^{-1})^{-1}$ and $\mathbf{M} = \alpha^{-1} \Sigma_q + \Sigma_x$.

The (latent) KL control problem can now be formulated using state cost $q(x_t)$ and uncontrolled dynamics \mathbf{P} as in Eq. (3). The optimal state transition $u^*(x_{t+1} | x_t)$ under controlled dynamics is given by Eq. (4).

4.3 CONTROL COMPUTATION IN OBSERVED SPACE

We are now ready to describe how to use latent KL control in the real system. Given an observation sequence $\mathbf{y}_{1:t}$ until time t , we can compute predictive distributions of the next observation \mathbf{y}_{t+1} under both the uncontrolled dynamics $p(x_{t+1} | x_t)$ and the optimally controlled dynamics $u^*(x_{t+1} | x_t)$ in the latent space as:

$$\begin{aligned} p(\mathbf{y}_{t+1} | \mathbf{y}_{1:t}) &= \sum_{x_{t:t+1}} p(\mathbf{y}_{t+1} | x_{t+1}) p(x_{t+1} | x_t) u(x_t | \mathbf{y}_{1:t}) \\ u(\mathbf{y}_{t+1} | \mathbf{y}_{1:t}) &= \sum_{x_{t:t+1}} p(\mathbf{y}_{t+1} | x_{t+1}) u^*(x_{t+1} | x_t) u(x_t | \mathbf{y}_{1:t}) \end{aligned}$$

where $u(x_t | \mathbf{y}_{1:t})$ denotes the filtered state at time t following the controlled process that evolves according to

$u^*(x'|x)$. Since we keep the previous filtered estimate $u(x_{t-1}|\mathbf{y}_{1:t-1})$, this computation is simply as

$$u(x_t|\mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_t|x_t) \sum_{x_{t-1}} u^*(x_t|x_{t-1}) u(x_{t-1}|\mathbf{y}_{1:t-1})}{u(\mathbf{y}_t|\mathbf{y}_{1:t-1})}.$$

We finally compute the control input command to the system such that the “difference” between the uncontrolled and optimal behaviors is reduced

$$\boldsymbol{\tau}_t = \mathbf{K}(\bar{\mathbf{y}}_{t+1|1:t}^u - \bar{\mathbf{y}}_{t+1|1:t}^p), \quad (6)$$

where $\bar{\mathbf{y}}_{t+1|1:t}^u$ and $\bar{\mathbf{y}}_{t+1|1:t}^p$ are the expectations of \mathbf{y} over $u(\mathbf{y}_{t+1}|\mathbf{y}_{1:t})$ and $p(\mathbf{y}_{t+1}|\mathbf{y}_{1:t})$ respectively and \mathbf{K} is a gain matrix to be tuned. The gain \mathbf{K} can be optimally computed if the model of system dynamics is available (Todorov, 2009b), however, in this paper we focus on the model-free scenario and leave it as a free parameter.

5 APPROXIMATE CONTROL USING FACTORIAL HIDDEN MARKOV MODELS

For high-dimensional problems that require to cover large regions of the state space, the previous approach becomes infeasible, since the cardinality required for the latent variable grows exponentially. In this section, we consider an alternative model with a multi-dimensional latent variable and constrained state transitions. We consider each dimension independent from the rest in the absence of control. These assumptions are naturally expressed using factorial HMMs. The advantage is that we can capture complex latent dynamics more efficiently. The price to pay is that exact optimal control computation in the latent space is no longer feasible and different approximation schemes have to be used. We describe this approach in the following sections.

5.1 FACTORIAL HIDDEN MARKOV MODELS

FHMM is a special type of HMM to model sequences of observations originated from multiple latent dynamical processes that interact to generate a single output (Ghahramani and Jordan, 1997; Murphy, 2012). The state is represented by a collection of variables $\mathbf{x}_t = \{x_t^{(1)}, \dots, x_t^{(m)}, \dots, x_t^{(M)}\}$ each of them having K possible values. The latent state \mathbf{x}_t is thus a M -dimensional variable with K^M possible values.

We will use a 1-of- K encoding, such that each state component $\mathbf{x}_t^{(m)}$ will be denoted using a $K \times 1$ vector, where each of the K discrete values corresponds to a 1 in one position and 0 elsewhere.

The assumption is that the transition model factorizes among the individual components

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \prod_{m=1}^M p^{(m)}(\mathbf{x}_t^{(m)}|\mathbf{x}_{t-1}^{(m)}), \quad (7)$$

where $p^{(m)}(\mathbf{x}_t^{(m)}|\mathbf{x}_{t-1}^{(m)})$ is the state transition matrix $\mathbf{P}^{(m)}$ for the m -th chain. We assume the Gaussian observation model, which is defined as

$$p(\mathbf{y}_t|\mathbf{x}_t) = \mathcal{N}\left(\sum_{m=1}^M \mathbf{W}^{(m)} \mathbf{x}_t^{(m)}, \Sigma\right) \quad (8)$$

where $\mathbf{W}^{(m)}$ is a $D \times K$ weight matrix that contains in its columns the contributions to the means for each of the possible configurations of $\mathbf{x}_t^{(m)}$. The marginal over \mathbf{y}_t is thus a Gaussian mixture model, with K^M Gaussian mixture components, each having a constant covariance matrix Σ .

The parameters $\boldsymbol{\theta}_{\text{FHMM}} = \{\mathbf{P}^{1:M}, \mathbf{W}^{1:M}, \Sigma\}$ can be learned using EM, as before. In this case, however, the E-step becomes intractable, since the forward-backward step has time complexity $O(TM K^{M+1})$. An alternative approximation that works well in practice is the structured mean field approximation, which has time complexity $O(TM K^2 I)$, where I is the number of mean field iterations (see Ghahramani and Jordan, 1997; Murphy, 2012, for details).

5.2 CONTROL COMPUTATION IN LATENT SPACE

In a similar way as in Section 4.2, we need first to define a cost function in the latent space $q(\mathbf{x}_t)$ to be able to formulate a KL control problem. A natural way to define $q(\mathbf{x}_t)$ given the observation model of Eq. (8) and the cost function in observation space $\tilde{q}(\mathbf{y}_t)$ is

$$q(\mathbf{x}_t) = \alpha \tilde{q}\left(\sum_{m=1}^M \mathbf{W}^{(m)} \mathbf{x}_t^{(m)}\right). \quad (9)$$

Computing the exact optimal control using Eq. (3) in FHMMs requires to transform the model into a single chain model with K^M states, which is intractable. We assume approximate controlled dynamics $u_{\text{ap}}(\mathbf{x}_t|\mathbf{x}_{t-1})$ and associated stationary distribution $\Pi_{\text{ap}}(\mathbf{x}_t)$ that factorize in its components:

$$u_{\text{ap}}(\mathbf{x}_t|\mathbf{x}_{t-1}) = \prod_{m=1}^M u_{\text{ap}}^{(m)}(\mathbf{x}_t^{(m)}|\mathbf{x}_{t-1}^{(m)})$$

$$\Pi_{\text{ap}}(\mathbf{x}_t) = \prod_{m=1}^M \Pi_{\text{ap}}^{(m)}(\mathbf{x}_t^{(m)}).$$

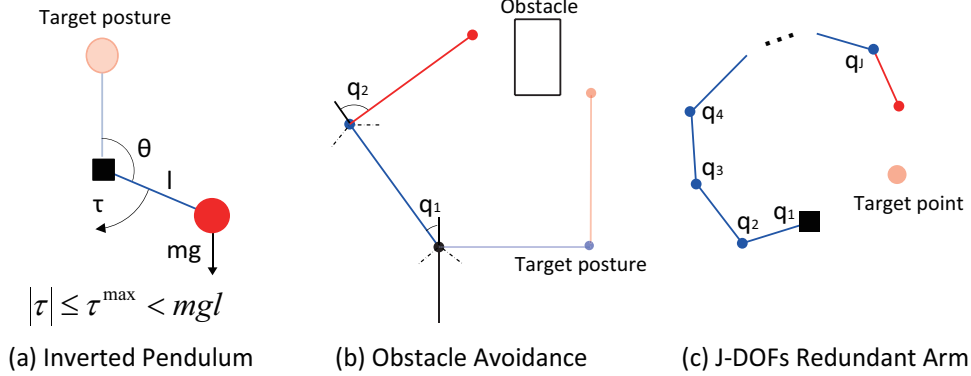


Figure 1: Motor control problems with simulated robots: **(a) Pendulum swing up with limited torque.** The state variable is $\mathbf{y} = [\theta, \omega]^T$ where $\omega = \dot{\theta}$, $|\theta| \leq \pi$, $|\omega| \leq 4\pi$. The control input is the torque at the joint τ . Uncontrolled dynamics and control matrix are given as $\mathbf{f}(\mathbf{y}) = [0 \ 1; g \sin(\theta)/l \ \mu/\omega]$, $\mathbf{B}(\mathbf{y}) = 1/ml^2$, respectively. Parameters values are $m = l = 1$, $g = 9.8$, $\mu = 0.25$ and $\tau^{\max} = 5.0$ that satisfies $\tau^{\max} < mgl$; **(b) Robot arm control with obstacle.** The state variable is $\mathbf{y} = [q_1, q_2]^T \in \mathcal{S}$ where \mathcal{S} is the state space that satisfies the joint angle limits and no collisions with the obstacle. The control input is $\boldsymbol{\tau} = \dot{\mathbf{y}}$. The uncontrolled dynamics and control matrix are $\mathbf{f}(\mathbf{y}) = [0, 0]^T$ and $\mathbf{B}(\mathbf{y}) = \mathbf{I}_D$; **(c) Multi-DOF redundant arm reaching task.** The state variable is $\mathbf{y}_t = [q_1(t), \dots, q_J(t)]^T$, $q_i(t) \in \mathcal{S}$ is the i -th joint angle and \mathcal{S} is the state space that satisfies the joint angle limit $-0.5\pi \leq q_i(t) \leq 0.5\pi$. The control input, uncontrolled dynamics and control matrix are as in (b), but for J dimensions. In all examples we use first-order Euler method for numerical integration.

These assumptions imply that the KL cost term can also be decomposed such that Eq. (1) becomes

$$\sum_{\mathbf{x}_t} \prod_{m=1}^M \Pi_{\text{ap}}^{(m)}(\mathbf{x}_t^{(m)}) \times \left(q(\mathbf{x}_t) + \sum_{m=1}^M \text{KL} \left(u_{\text{ap}}^{(m)}(\cdot | \mathbf{x}_t^{(m)}) \parallel p^{(m)}(\cdot | \mathbf{x}_t^{(m)}) \right) \right). \quad (10)$$

We can minimize Eq. (10) iteratively using sequential updates: for each chain m , update the parameters $u_{\text{ap}}^{(m)}$ and $\Pi_{\text{ap}}^{(m)}$ assuming the parameters for the other chains fixed so that it minimizes the marginal state-dependent cost

$$Q^{(m)}(\mathbf{x}_t^{(m)}) = \sum_{\mathbf{x}_t^{(i)}, i \neq m} \prod_{i \neq m} \Pi_{\text{ap}}^{(i)}(\mathbf{x}_t^{(i)}) q(\mathbf{x}_t) \quad (11)$$

and the corresponding KL cost. Each update corresponds to a sub-problem of the type of Eq. (3) and can be solved as a principal eigenvector problem. The average cost monotonically decreases at each iteration and its convergence is guaranteed. We call this scheme *Variational KL minimization (VKL)*.

Note however, **VKL** requires summing over all the values of the $M - 1$ chains to obtain the marginal state-dependent cost, and thus it has time complexity $O(K^{M-1})$, which is still intractable. We further

approximate this computation by taking the expected state of the other chains according to their individual stationary distributions

$$Q^{(m)}(\mathbf{x}_t^{(m)}) \approx \alpha \tilde{q} \left(\mathbf{W}^{(m)} \mathbf{x}_t^{(m)} + \sum_{i \neq m} \mathbf{W}^{(i)} \Pi_{\text{ap}}^{(i)} \right), \quad (12)$$

where $\Pi_{\text{ap}}^{(i)}$ is a K -dimensional vector with the stationary distribution of chain i . Evaluation of Eq. (12) only requires $O(KM)$ steps, and it is therefore tractable. We refer this approximation as *Approximate Variational KL minimization (AVKL)*.

We refer to the control computed using either **VKL** and **AVKL** as u_{ap}^* in the rest of this section.

5.3 CONTROL COMPUTATION IN OBSERVED SPACE

Having approximated our optimal control law in the latent space, we need to define a control law for the real (observed) system given sequence of observations $\mathbf{y}_{1:t}$. We follow the same approach as in Section 4.3. First, we obtain estimates for the expected values of the next observed state under both controlled and uncontrolled dynamics as $\bar{\mathbf{y}}_{t+1|1:t}^u$ and $\bar{\mathbf{y}}_{t+1|1:t}^p$, respectively. Second, we apply the controller of Eq. (6).

The first step requires to solve a filtering problem to obtain $u(\mathbf{x}_t|\mathbf{y}_{1:t})$, which is intractable for this model. We use an approximate approach based on structured mean field, as in the model learning step (Section 5.1, E-step). However, instead of keeping the last filtered estimate $u(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})$ as before, we keep the filtered estimate at time-step $t-H$, i.e. $u(\mathbf{x}_{t-H}|\mathbf{y}_{1:t-H})$ and perform *offline* structured mean field using the last H observations $\mathbf{y}_{t-H:t}$. This approach improves considerably the accuracy of the filtered estimates $u(\mathbf{x}_t|\mathbf{y}_{1:t}) = \prod_m u^{(m)}(\mathbf{x}_t^{(m)}|\mathbf{y}_{1:t})$ and at the same time, it is more efficient than structured mean field on the entire sequence of past observations.

Once we have filtered estimates of the latent state, the expectation of \mathbf{y}_{t+1} over predictive distribution $u(\mathbf{y}_{t+1}|\mathbf{y}_{1:t})$ can be approximated using samples

$$\begin{aligned}\bar{\mathbf{y}}_{t+1|1:t}^u &= \int \mathbf{y}_{t+1} u(\mathbf{y}_{t+1}|\mathbf{y}_{1:t}) d\mathbf{y}_{t+1} \\ &\approx \frac{1}{L} \sum_{\mu=1}^L \sum_{m=1}^M \mathbf{W}^{(m)} \hat{\mathbf{x}}_{\mu}^{(m)}\end{aligned}$$

where $\hat{\mathbf{x}}_{\mu}^{(m)}$ are samples drawn from the posterior distribution of the latent component according to the approximated controlled dynamics

$$\begin{aligned}\hat{\mathbf{x}}_{\mu}^{(m)} &\sim u^{(m)}(\mathbf{x}_{t+1}^{(m)}|\mathbf{y}_{1:t}) \\ &= \sum_{\mathbf{x}_t^{(m)}} u_{\text{ap}}^{*,(m)}(\mathbf{x}_{t+1}^{(m)}|\mathbf{x}_t^{(m)}) u^{(m)}(\mathbf{x}_t^{(m)}|\mathbf{y}_{1:t}).\end{aligned}\quad (13)$$

Similarly, we can estimate $\bar{\mathbf{y}}_{t+1|1:t}^p$ using samples from

$$p^{(m)}(\mathbf{x}_{t+1}^{(m)}|\mathbf{y}_{1:t}) = \sum_{\mathbf{x}_t^{(m)}} p^{(m)}(\mathbf{x}_{t+1}^{(m)}|\mathbf{x}_t^{(m)}) u^{(m)}(\mathbf{x}_t^{(m)}|\mathbf{y}_{1:t}).$$

We show in the next section that for relatively small values of the window length H and the number of samples L , the resulting controls are satisfactory.

6 SIMULATION RESULTS

In this section, we apply our method to three benchmark (simulated) robot motor control problems: (a) pendulum swing-up with limited torque (Doya, 2000), (b) robot arm control with obstacles (Sugiyama et al., 2007), and (c) multi-degrees of freedom (DOF) redundant arm reaching task (Theodorou et al., 2010). Figure 1 illustrates these problems. The first two examples correspond to the approach using HMMs of Section 4 whereas the third shows an application using FHMMs as described in Section 5.

For learning the HMM parameters, we use identical and independent exploration noise in all controlled dimensions parameterized by σ_{ϵ}^2 , i.e. $(\Sigma_{\epsilon})_{ij} = \delta_{ij}\sigma_{\epsilon}^2$.

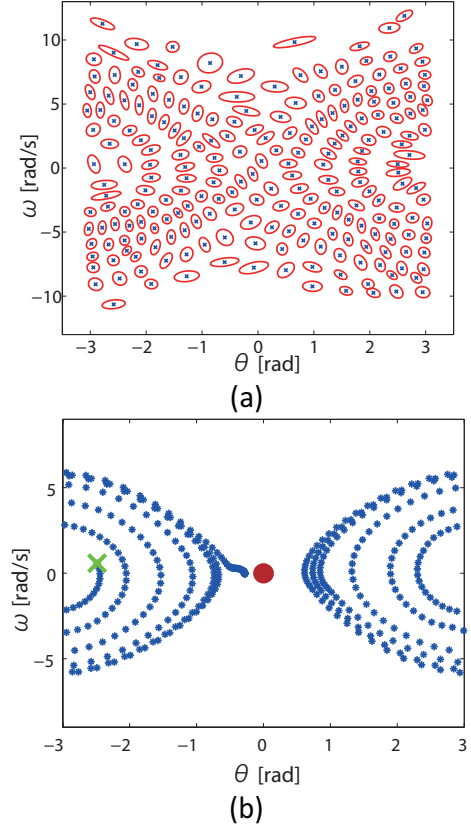


Figure 2: Pendulum swing-up task results: **(a)** Observation model after learning the HMM with $N = 225$ hidden states and $\sigma_{\epsilon} = 1.5$. Each hidden state corresponds to a two-dimensional Gaussian distribution with mean indicated by a cross and contour with equal probability density shown as an ellipse. **(b)** Typical controlled behaviour in the phase plane. The cross and the circle show initial and target states respectively.

Both tasks consider a two-dimensional observed continuous state and a one-dimensional latent variable. The complexity of the method strongly depends on the number of hidden values N . For this experiments, we simply choose N large enough ($N = 255$ in both scenarios) to obtain a model that accurately describes the system dynamics. We learn the full parameter vector $\boldsymbol{\theta}_{\text{HMM}}$ using EM with K-means initialization for the Gaussian means.

6.1 PENDULUM SWING-UP TASK

This is a non-trivial problem when the maximum torque τ^{\max} is smaller than the maximal load torque mgL . The optimal control requires to take an energy-efficient strategy: swing the pendulum several times to build up momentum and also decelerate the pendulum early enough to prevent it from falling over.

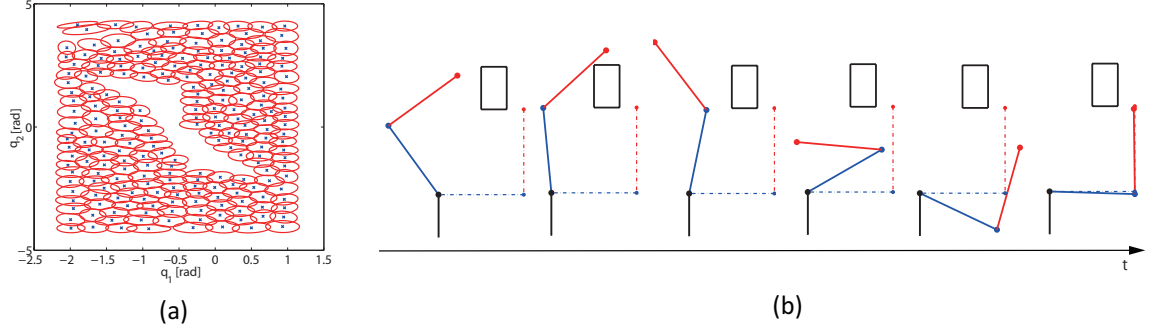


Figure 3: Results on the robot arm with an obstacle. **(a)** Learned HMM with $N = 225$, $\Sigma_\epsilon = \text{diag}\{1.5, 1.5\}$ and $T = 3 \cdot 10^4$ samples. **(b)** Controlled robot arm behavior at different time steps. The robot successfully reaches the target posture avoiding the obstacle.

Fig. 2(a) shows the 2-dimensional observation model after learning with exploration noise $\sigma_\epsilon^2 = 1.5$. We can see that the HMM is able to capture a discrete, coarse representation of the continuous state.

For control computation, we define a quadratic cost $\tilde{q}(\mathbf{y}_t) = \mathbf{y}_t^T \Sigma_q^{-1} \mathbf{y}_t$, where $\Sigma_q = \text{diag}\{0.005, 0.02\}$, and set the scale parameter $\alpha = \alpha_0 \Delta t / \sigma_\epsilon^2$ to prevent the scaling effect of the exploration noise variance σ_ϵ^2 in the KL cost ($\alpha_0 = 0.2$). The gain matrix is $\mathbf{K} = \text{diag}\{50, 10\}$. The eigenvector computation only takes $3 \cdot 10^{-2}$ seconds¹. The computation of control input (see Section 4.3) takes $3 \cdot 10^{-3}$ seconds per time-step. The resulting controller successfully maintains the pendulum in a region of $|\theta| \leq 0.5$ continuously in all tested random initializations and it is optimal in terms of energy-efficiency. A typical controlled behavior of the pendulum is shown in Fig. 2(b).

For comparison, we also implemented standard value iteration (VI) (Sutton and Barto, 1998), which requires knowledge of the true pendulum dynamics and uses a fully discretized state-action space. For consistency, we choose as a cost function $r(\mathbf{y}, \mathbf{u}) = \alpha \tilde{q}(\mathbf{y}_t) + \frac{1}{2} \|\mathbf{u}\|^2$ and the same error tolerance 10^{-8} for both value iteration method and power method. VI requires a very fine discretization ($N \geq 1225$ states) and at least 20 seconds of CPU-time, which are roughly an order of magnitude larger than the values obtained using the proposed method.

6.2 ROBOT ARM CONTROL WITH OBSTACLE

In this second task, we aim to control a two-joint robot arm from an initial posture to the target posture while avoiding an obstacle. The presence of the obstacle

makes this task difficult to solve using standard trajectory interpolation methods, see Fig. 1(b) for details.

Fig. 3(a) shows the 2D observation model learned using the same setup as before. As the empty region in the middle of the plot indicates, the model successfully captures the physically impossible state transitions that would bring the robot arm through the obstacle.

For this problem, we set the cost function as $\tilde{q}(\mathbf{y}_t) = (\mathbf{y} - \mathbf{g})^T \Sigma_q^{-1} (\mathbf{y} - \mathbf{g})$, where $\Sigma_q = \text{diag}\{0.01, 0.01\}$ and $\mathbf{g} = [-\pi/2, \pi/2]^T$. In this case, we use $\alpha_0 = 0.05$ and $\mathbf{K} = \text{diag}\{3.0, 0.5\}$ to set the scale parameter and the gain matrix, respectively. Computation time of the optimal control is approximately 0.03 seconds using the same specifications as in the previous example. Fig. 3(b) illustrates the typical controlled robot behavior. The robot arm first decreases the angle q_2 and then modifies q_1 reaching the target posture while successfully avoiding the obstacle.

6.3 REACHING TASK

The third task consists of a multi-DOF planar robot arm with J joints and joint-limit constraints as shown in Fig. 1(c). The J joints are of equal length $l = 1$ and connected to a fixed base. Each joint dynamics of this robot model is decoupled, and therefore suitable for our method using FHMMs.

The goal is to control the joint angles to reach a target position $\mathbf{t}^{\text{target}}$ with the end-effector of the robot arm. For $J \gg 2$ the control policy has to make a choice among many possible trajectories in the joint space. Moreover, considering joint-limit constraints limits direct application of standard methods for inverse kinematic, e.g. Jacobian inverse techniques (Yoshikawa,

¹Core-i7 2.8GHz-CPU, 8GB memory and MATLAB.

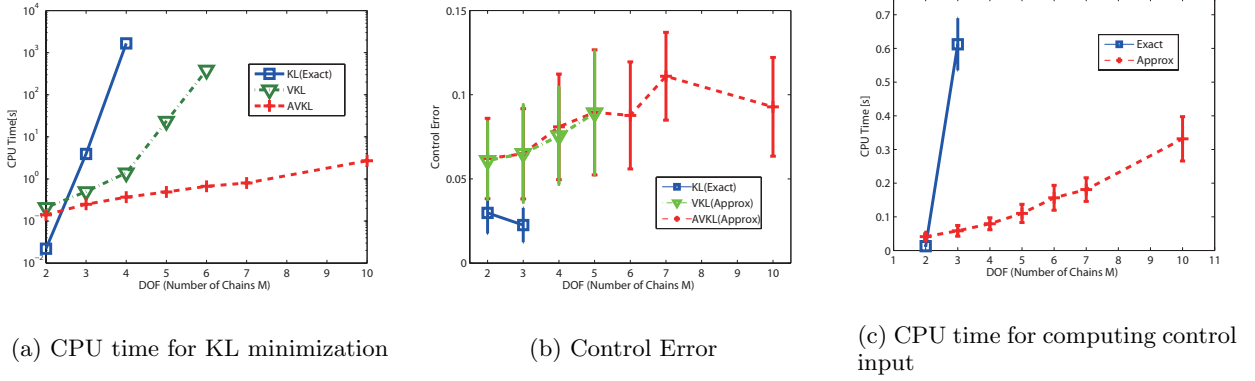


Figure 4: **Multi-DOF robot reaching task:** Comparison between **KL(exact)**, **VKL** and **AVKL**. **VKL** and **AVKL** can efficiently compute near optimal controller comparable to exact KL minimization. **AVKL** scales to high-dimensional problems. **KL(exact)** and **VKL** are only feasible for $J < 4$ and $J < 6$, respectively.

1990). The cost function for this task is

$$\tilde{q}(\mathbf{y}) = \|\mathbf{t}^{\text{target}} - \mathbf{T}(\mathbf{y})\|, \quad (14)$$

where $\mathbf{T}(\cdot)$ is the forward kinematics model that maps a joint angle vector to the corresponding end-effector position in the task space

$$\mathbf{T}(\mathbf{y}) = \begin{bmatrix} \sum_{n=1}^J \cos\left(\sum_{j=1}^n y_j\right) \\ \sum_{n=1}^J \sin\left(\sum_{j=1}^n y_j\right) \end{bmatrix}.$$

Although the dynamics decouples for each joint, the cost function couples all the joint angles making the problem difficult.

We analyze the scaling properties with the number J of degrees of freedom, comparing the different strategies described in Section 5.2: **KL(exact)** minimization, **VKL** and **AVKL**. The exact solution uses K^M states and performs exact inference. For approximate methods, we use as many latent dimensions (chains) as joints $M = J$, with $K = 20$ and $H = 2J$ time-steps for approximate filtering. Note that M could be smaller than J , as long as the learned hidden representation captures well the underlying structure and dynamics. We set $M = J$ to simplify the evaluation.

Convergence of variational eigen-computations **VKL** and **AVKL** is reached after approximately 10 iterations in this task (each iteration requires an update of all the parameters of the J joints). Learning the parameters of the FHMM is sensitive to local minima. In practice, we choose $\mathbf{W}^{(m)}$ so that each factored state represents each joint dynamics and only learn the uncontrolled dynamics (transition probabilities). Also, $\mathbf{t}^{\text{target}}$ is set to one of the $\mathbf{w}_i^{(m)}$ to prevent space quantization errors in this comparison.

Fig. 4 illustrates the comparison. Whereas **KL(exact)** and **VKL** are only feasible for $J < 5$ and $J < 7$ respectively, **AVKL** is applicable to a larger number of joints. Fig. 4(a) shows CPU-time for control computation in the latent space (Section 5.2), which scales exponentially for both **KL(exact)** and **VKL** and approximately linear for **AVKL**.

Fig. 4(b) shows the error Eq. (14) averaged over 200 trials with randomly initialized joints. Although exact control computation can be performed for $M < 5$, exact inference is only possible for $M < 4$. We can observe that the resulting controls are satisfactory and errors do not differ significantly between **VKL** and **AVKL**. Notice that the **AVKL** error remains approximately constant as a function of M .

Fig. 4(c) shows CPU-time for the control computation in the observed space (Section 5.3). While CPU-time for exact computation quickly increases, our approximate approach results in a roughly linear increase.

Examples of controlled robot behaviors for a different number of degrees of freedom are shown in Fig. 5. In all cases, the robot successfully reaches the goal while satisfying the joint-limit constraints starting from several initial postures.

From these results we can conclude that it is feasible to learn FHMMs for high-dimensional systems with uncoupled uncontrolled dynamics and that latent KL control is an effective method to near-optimally control such systems.

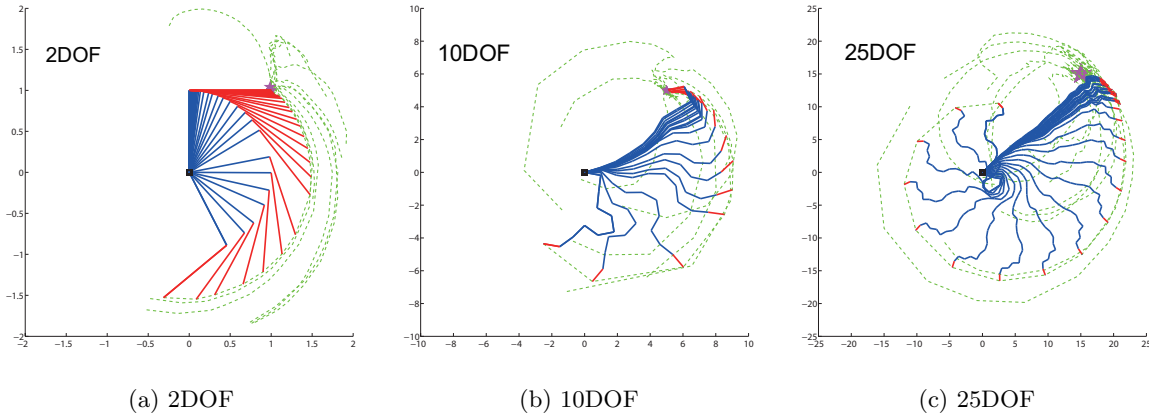


Figure 5: **Multi-DOF robot reaching task:** Examples of robot trajectories. The arm successfully reaches the target position while satisfying the joint-limit constraints from several initializations. Green lines show end-effector trajectories for different initializations. Blue and red lines indicate intermediate and end links.

7 DISCUSSION

We have proposed a novel solution that combines the KL control framework with probabilistic graphical models in the infinite horizon, average cost setting. Our approach learns a coarse, discrete representation amenable for efficient computation to near-optimally control continuous-state systems. We have presented two examples, using hidden Markov models (HMMs) and factorial HMMs (FHMMs), and we have shown evidence that our proposed method is feasible in three robotic tasks. In particular, we have demonstrated that the second example with FHMMs is scalable to higher dimensional problems.

The presented latent KL control approach (with HMMs) resembles the one of Zhong and Todorov (2011a) which considers an “aggregated” space similar to the latent space of the HMM. However, note that whereas for Zhong and Todorov (2011a) the real model is *required in the observed space*, in our case we *learn an approximate model* in which observations are coupled through the latent variables. Their main computational bottleneck is the “double” numerical integration over the observed space for computing the “aggregated” state transition probability. In our case, we replace such a problem by a probabilistic graphical model learning problem.

The control performance strongly depends on the quality of the learned model, which requires choosing a proper exploration noise and a proper initialization of the graphical model parameters. Current work is focused in alternative learning methods that efficiently sample interesting regions of the state space and exploit the ergodic nature of the problems. Extension to more complex scenarios is also being considered.

ACKNOWLEDGEMENT

This work was supported by JSPS KAKENHI Grant Number 25540085 and by the European Community Seventh Framework Programme (FP7/2007-2013) under grant agreement 270327 (CompLACS).

References

- Doya, K. (2000). Reinforcement learning in continuous time and space. *Neural Comput.*, 12(1):219–245.
- Ghahramani, Z. and Jordan, M. (1997). Factorial hidden markov models. *Mach. Learn.*, 29(2-3):245–273.
- Kappen, H. J., Gómez, V., and Opper, M. (2012). Optimal control as a graphical model inference problem. *Mach. Learn.*, 87(2):159–182.
- Kinjo, K., Uchibe, E., and Doya, K. (2013). Evaluation of linearly solvable Markov decision process with dynamic model learning in a mobile robot navigation task. *Front. Neurobot.*, 7:1–13.
- Mitrovic, D., Nagashima, S., Klanke, S., Matsubara, T., and Vijayakumar, S. (2010). Optimal feedback control for anthropomorphic manipulators. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA’10)*, pages 4143–4150.
- Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press.
- Sugiyama, M., Hachiya, H., Towell, C., and Vijayakumar, S. (2007). Value function approximation on non-linear manifolds for robot motor control. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA’07)*, pages 1733–1740.
- Sutton, R. S. and Barto, A. G. (1998). *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition.
- Theodorou, E., Buchli, J., and Schaal, S. (2010). Reinforcement learning of motor skills in high dimensions: A path integral approach. In *Proceedings of the IEEE*

International Conference on Robotics and Automation (ICRA '10), pages 2397–2403.

Todorov, E. (2006). Linearly-solvable markov decision problems. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1369–1376.

Todorov, E. (2009a). Efficient computation of optimal actions. *PNAS*, 106(28):11478–11483.

Todorov, E. (2009b). Eigenfunction approximation methods for linearly-solvable optimal control problems. In *Proceedings of the 2nd IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, pages 161–168.

Yoshikawa, T. (1990). *Foundations of Robotics: Analysis and Control*. The MIT Press.

Zhong, M. and Todorov, E. (2011a). Aggregation methods for linearly-solvable Markov decision process. In *World Congress of the International Federation of Automatic Control*, pages 11220–11225.

Zhong, M. and Todorov, E. (2011b). Moving least-squares approximations for linearly-solvable stochastic optimal control problems. *J. Control. Theory. Appl.*, 9(3):451–463.

GPS-ABC: Gaussian Process Surrogate Approximate Bayesian Computation

Edward Meeds
Informatics Institute
University of Amsterdam
tmeeds@gmail.com

Max Welling
Informatics Institute
University of Amsterdam
welling.max@gmail.com

Abstract

Scientists often express their understanding of the world through a computationally demanding simulation program. Analyzing the posterior distribution of the parameters given observations (the inverse problem) can be extremely challenging. The Approximate Bayesian Computation (ABC) framework is the standard statistical tool to handle these likelihood free problems, but they require a very large number of simulations. In this work we develop two new ABC sampling algorithms that significantly reduce the number of simulations necessary for posterior inference. Both algorithms use confidence estimates for the accept probability in the Metropolis Hastings step to adaptively choose the number of necessary simulations. Our GPS-ABC algorithm stores the information obtained from every simulation in a Gaussian process which acts as a surrogate function for the simulated statistics. Experiments on a challenging realistic biological problem illustrate the potential of these algorithms.

1 Introduction

The morphogenesis of complex biological systems, the birth of neutrons stars, and weather forecasting are all natural phenomena whose understanding relies deeply upon the interaction between simulations of their underlying processes and their naturally observed data. Hypotheses that posit the generation of observations evolve after critical evaluation of the match between simulation and observation.

This hypothesis–simulation–evaluation cycle is the foundation of *simulation-based modeling*. For all but the most trivial phenomena, this cycle is grossly inefficient. A typical simulator is a complex computer program with a potentially large number of interacting parameters that not

only drive the simulation program, but are also often the variables of scientific interest because they play a role in explaining the natural phenomena. Choosing an optimal value setting for these parameters can be immensely expensive.

While a single, optimal parameter setting may be useful to scientists, often they are more interested in the *distribution* of parameter settings that provide accurate simulation results [20, 3, 18]. The interaction between parameters can provide insight regarding not only the properties of the simulation program, but more importantly, the underlying phenomena of interest. The main challenges that we address in this paper are 1) simulation-based modeling in a *likelihood-free* setting (we do not have a model in the typical machine learning sense, and therefore we do not have a standard likelihood function), and 2) running simulations is very expensive.

The first challenge is partly addressed by the *approximate Bayesian computation* (ABC) approach to sampling in likelihood-free scenarios [25, 5]. The ABC approach will be described in a later section, but in short it uses the distance between simulated and observed data as a proxy for the likelihood term in the parameter posterior. ABC provides the necessary framework to make progress in simulation-based modeling, but it is a very inefficient approach, even on simple problems.

The second challenge is approached with a surrogate model in mind. This means that every simulation (parameters and result) is stored and used to maintain a surrogate of the mapping from parameters to result. By carefully constructing an approximate Markov chain Monte Carlo (MCMC) sampler, we are able to sample from the parameter distribution in such a way that our sampling error is controlled and decreases over time. The main advantage of this approach is that by accepting some bias, we are able to very efficiently sample from the approximate posterior because parameters can sometimes be accepted within MCMC with high confidence by relying on the surrogate and thus avoiding expensive simulations.

In this paper we present a procedure for approximate Bayesian inference using a Gaussian process surrogate for expensive simulation-based models. In our approach, simulations that are run over the course of the inference procedure are incorporated into a GP model, gradually improving the surrogate over time. Using MCMC with a Metropolis-Hastings (MH) accept/reject rule, our method uses the surrogate and its uncertainty to make all MH decisions. The key insight is that the uncertainty in the acceptance probability is used to determine if simulations are required to confidently proceed to the MH step. If the uncertainty is high, and we are likely to make an error, then more simulations are run.

2 Approximate Bayesian Computation

The current state-of-the-art for simulation-based model inference is *likelihood-free* or *approximate Bayesian computation* methods [23, 15]. In this section we briefly introduce likelihood-free inference with a focus on MCMC inference procedures as our modeling approach will extend naturally from this work.

In likelihood-free sampling, we do not have a model in the typical sense. Instead, we have access to a simulator that generates pseudo-data that, given an accurate model of the world, look like the observations. The goal is to infer the parameters of the simulator which produce accurate pseudo-data. Importantly, we do not have access to a tractable likelihood function. We now describe in detail the likelihood-free set-up and in particular MCMC sampling techniques.

One of the primary goals of Bayesian inference is to infer the posterior distribution of latent variables θ using its prior $\pi(\theta)$ and its data likelihood $\pi(\mathbf{y}|\theta)$:

$$\pi(\theta|\mathbf{y}) = \frac{\pi(\theta)\pi(\mathbf{y}|\theta)}{\int \pi(\theta)\pi(\mathbf{y}|\theta)d\theta} \quad (1)$$

where θ is a vector of latent parameters and \mathbf{y} is the observed data set. In simulation-based modeling, we do not have access to the likelihood $\pi(\mathbf{y}|\theta)$. Instead our model of the world consists of a simulator that generates samples $\mathbf{x} \stackrel{\text{sim}}{\sim} \pi(\mathbf{x}|\theta)$ (where we indicate that the simulator was run with parameters θ and returns pseudo-data \mathbf{x}). Simulation results \mathbf{x} are then compared with the observations \mathbf{y} through a distribution $\pi_\epsilon(\mathbf{y}|\mathbf{x}, \theta)$, which measures how similar \mathbf{x} is to \mathbf{y} . The distribution is parameterized by ϵ which controls the acceptable discrepancy between \mathbf{x} and \mathbf{y} . We can thus approximately infer the posterior distribution as

$$\pi_\epsilon(\theta|\mathbf{y}) = \frac{\pi(\theta)}{\pi(\mathbf{y})} \int \pi_\epsilon(\mathbf{y}|\mathbf{x})\pi(\mathbf{x}|\theta)d\mathbf{x} \quad (2)$$

This approximate posterior is only equal to the true posterior for $\pi_{\epsilon=0}(\mathbf{y}|\mathbf{x}) = \delta(\mathbf{y}, \mathbf{x})$ where $\delta(\cdot)$ is the delta-

function. For the exact posterior, one could apply a rejection sampling procedure that would repeatedly sample $\theta \sim \pi(\theta)$, run a simulation $\mathbf{x} \stackrel{\text{sim}}{\sim} \pi(\mathbf{x}|\theta)$, then accept θ only if it equals \mathbf{y} . For continuous data, ϵ acts as a slack variable because equality cannot be achieved. However, we prefer small ϵ because this will improve our approximation to the true posterior. Unfortunately, there remains an unavoidable trade-off between approximation bias (large for large ϵ) and rejection rate (large for small ϵ).

2.1 Marginal and Pseudo-Marginal ABC

Instead of the rejection sampler described in the previous section (which is hopeless in high dimensions), we now describe two MCMC procedures, the marginal sampler and the pseudo-marginal sampler [2]. At every iteration of MCMC we propose a new $\theta' \sim q(\theta'|\theta)$. Next we generate S samples $\mathbf{x}'_s \stackrel{\text{sim}}{\sim} \pi(\mathbf{x}'|\theta')$, $s = 1..S$ from the simulator. From these samples we approximate the marginal likelihood as follows,

$$\pi_\epsilon(\mathbf{y}|\theta') = \int \pi_\epsilon(\mathbf{y}|\mathbf{x})\pi(\mathbf{x}|\theta')d\mathbf{x} \approx \frac{1}{S} \sum_{s=1}^S \pi_\epsilon(\mathbf{y}|\mathbf{x}^{(s)}, \theta') \quad (3)$$

We accept the proposed parameter θ' with probability equal to,

$$\alpha(\theta'|\theta) = \min \left(1, \frac{\pi(\theta') \sum_s \pi_\epsilon(\mathbf{y}|\mathbf{x}'^{(s)}, \theta') q(\theta|\theta')}{\pi(\theta) \sum_s \pi_\epsilon(\mathbf{y}|\mathbf{x}^{(s)}, \theta) q(\theta'|\theta)} \right) \quad (4)$$

where the estimate of the marginal likelihood in the denominator (based on $\{\mathbf{x}_s, \theta\}$) is carried over from the previous iteration. It can be shown that this algorithm is an instance of the more general pseudo-marginal procedure [2] because the estimate of the marginal likelihood is unbiased. From this we can immediately infer that this Markov chain converges to the posterior $\pi_\epsilon(\theta|\mathbf{y})$. Interestingly, there is an alternative view of this sampler [22] that interprets it as a Markov chain over the extended state $\{\theta, \mathbf{x}_1, \dots, \mathbf{x}_S\}$ which also leads to the conclusion that the samples θ' will asymptotically follow the distribution $\pi_\epsilon(\theta|\mathbf{y})$.

Unfortunately, it is well known that pseudo-marginal samplers can suffer from slow mixing. In particular, when through a “lucky” draw our marginal likelihood estimate in Eqn. 3 attains a large value, then it is very difficult for the sampler to mix away from that state. To avoid this behavior it is sometimes beneficial to re-estimate the denominator (as well as the numerator) in every iteration. This procedure is more expensive and does not guarantee convergence to $\pi_\epsilon(\theta|\mathbf{y})$ (unless $S \rightarrow \infty$), but can result in much better mixing. We will call this the marginal LF MCMC method [2].

While for the pseudo-marginal approach we can interpret the fluctuations induced by estimating the $\pi(\mathbf{y}|\theta)$ from a

finite sample set as part of randomly proposing a new state, this is no longer true for the approximate marginal MCMC. For the latter it is instructive to study the uncertainty in the acceptance probability $\alpha(\theta'|\theta)$ due to these fluctuations: repeatedly estimating $\pi(\mathbf{y}|\theta)$ with S samples will produce a distribution over α . Clearly for small S the distribution will be wider while for very large S it will approach a delta peak. Our approach uses this distribution directly to determine the confidence in the MH accept step, allowing it to implicitly set S based on the local uncertainty. This will be discussed further in next sections.

Besides the marginal and pseudo-marginal approaches to ABC, there is a large body of work using sequential Monte Carlo sampling to approach this problem [21, 4, 26].

3 The Synthetic Likelihood

We next discuss the approximation introduced by Wood [29], who models the simulated pseudo-data $\{\mathbf{x}_1, \dots, \mathbf{x}_S\}$ at θ using a normal distribution, i.e. $\pi(\mathbf{x}|\theta) \approx \mathcal{N}(\mathbf{x}|\hat{\mu}_\theta, \hat{\Sigma}_\theta)$. We will later replace this with a Gaussian process. The procedure is very simple: we draw S samples from our simulator and compute first and second order statistics,

$$\hat{\mu}_\theta = \frac{1}{S} \sum_s \mathbf{x}^{(s)} \quad (5)$$

$$\hat{\Sigma}_\theta = \frac{1}{S-1} \sum_s (\mathbf{x}^{(s)} - \hat{\mu}_\theta) (\mathbf{x}^{(s)} - \hat{\mu}_\theta)^T \quad (6)$$

Using estimators $\hat{\mu}_\theta$ and $\hat{\Sigma}_\theta$ we set $\pi(\mathbf{x}|\theta) = \mathcal{N}(\hat{\mu}_\theta, \hat{\Sigma}_\theta)$. Moreover, if we use a Gaussian kernel, then

$$\pi_\epsilon(\mathbf{y}|\mathbf{x}) = K_\epsilon(\mathbf{y}, \mathbf{x}) = \frac{1}{(2\pi\epsilon)^{J/2}} e^{-\frac{1}{2\epsilon^2}(\mathbf{x}-\mathbf{y})^T(\mathbf{x}-\mathbf{y})} \quad (7)$$

where J is the dimension of \mathbf{y} , we can then analytically integrate over \mathbf{x} in Eqn 2 giving the *synthetic-ABC* likelihood:

$$\begin{aligned} \pi(\mathbf{y}|\theta) &= \int K_\epsilon(\mathbf{y}, \mathbf{x}) \mathcal{N}(\hat{\mu}_\theta, \hat{\Sigma}_\theta) d\mathbf{x} \\ &= \mathcal{N}(\hat{\mu}_\theta, \hat{\Sigma}_\theta + \epsilon^2 \mathbf{I}) \end{aligned} \quad (8) \quad (9)$$

which has the satisfying result that likelihood of $\mathbf{y}|\theta$ is the density under a Gaussian model at each simulation parameter setting θ .

This approximation has two advantages. First, we can take the limit $\epsilon \rightarrow 0$.¹ This implies that the bias introduced by the need to use a distribution $\pi_\epsilon(\mathbf{y}|\theta)$ to measure the similarity between simulations \mathbf{x} and observations \mathbf{y} is now

¹We may not always want to do this as ϵ^2 acts both as a prior over and a smoother of the likelihood. In practice, smoothing the likelihood may be necessary for mixing, and likewise, we may have access to a prior which could make the sampler more robust.

Algorithm 1 Synthetic-likelihood ABC MH step

inputs: $q, \theta, \pi(\mathbf{x}|\theta), S, \epsilon, \mathbf{y}$
 $\theta' \sim q(\theta'|\theta)$
for $s = 1 : S$ **do**
 $\mathbf{x}'^{(s)} \stackrel{\text{sim}}{\sim} \pi(\mathbf{x}|\theta'), \mathbf{x}^{(s)} \stackrel{\text{sim}}{\sim} \pi(\mathbf{x}|\theta)$
end for
Set $\hat{\mu}_{\theta'}, \hat{\Sigma}_{\theta'}, \hat{\mu}_\theta, \hat{\Sigma}_\theta$ using Eqns 5 and 6.
Set α using Eqn 10
if $\mathcal{U}(0, 1) \leq \alpha$ **then**
return θ'
end if
return θ

removed. But this is traded off with the bias introduced by modeling the simulations from $\pi(\mathbf{x}|\theta)$ with a normal distribution. The second advantage was the main motivation in [29], namely that this procedure is more robust for extremely irregular probability distributions as encountered in chaotic or near chaotic simulation dynamics.

A marginal sampler based on a Gaussian approximation (Algorithm 1) has the following acceptance probability:

$$\alpha(\theta'|\theta) = \min \left(1, \frac{\pi(\theta') \mathcal{N}(\hat{\mu}_{\theta'}, \hat{\Sigma}_{\theta'} + \epsilon^2 \mathbf{I}) q(\theta|\theta')}{\pi(\theta) \mathcal{N}(\hat{\mu}_\theta, \hat{\Sigma}_\theta + \epsilon^2 \mathbf{I}) q(\theta'|\theta)} \right) \quad (10)$$

As with the marginal sampler of Section 2, the fact that we estimate first and second order statistics from a finite sample set introduces uncertainty in the accept probability $\alpha(\theta'|\theta)$: another run of S simulations would have resulted in different values for these statistics and hence of the accept probability. See Figure 1 for an illustration. In the following section we will analyze the distribution over $\alpha(\theta'|\theta)$ and develop a method to decide how many simulations S we need in order to be sufficiently confident that we are making the correct accept/reject decision. Random acceptance probability distributions have been studied in general [16] and for the specific case of Gaussian log-energy estimates [9].

3.1 MCMC with a Random Acceptance Probability

We now make explicit the role of randomness in the MCMC sampler with synthetic (normal) likelihoods. At each iteration of the MCMC sampler, we compute estimators $\{\hat{\mu}_\theta, \hat{\Sigma}_\theta, \hat{\mu}_{\theta'}, \hat{\Sigma}_{\theta'}\}$ as before using Eqns 5 and 6. To estimate the distribution over accept probabilities we would need M sets of S simulations, which would be too expensive. Instead, we use our Gaussian assumption to derive that the variance of the mean is $1/S$ times the variance in the sample $\{\mathbf{x}_1, \dots, \mathbf{x}_S\}$,

$$\mu_\theta \sim \mathcal{N}(\hat{\mu}_\theta, \hat{\Sigma}_\theta/S) \quad (11)$$

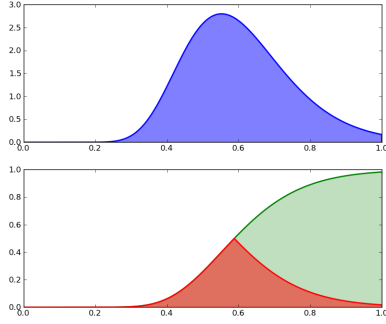


Figure 1: An example of $p(\alpha)$, the distribution over acceptance probabilities (**top**) and its CDF shown folded at its median (**bottom**).

and similarly for $\mu_{\theta'}$. This shortcut is important because it allows us to avoid a significant number of expensive simulations and replace them with samples from a normal distribution.

Given our M samples of $(\mu_{\theta}, \mu_{\theta'})$, we can compute M samples of $\alpha(\theta'|\theta)$ by inserting them into the expression for the randomized MH accept probability:

$$\alpha^{(m)} = \min \left(1, \frac{\pi(\theta') \mathcal{N}(\mathbf{y} | \mu_{\theta'}^{(m)}, \hat{\Sigma}_{\theta'} + \epsilon^2 \mathbf{I}) q(\theta|\theta')}{\pi(\theta) \mathcal{N}(\mathbf{y} | \mu_{\theta}^{(m)}, \hat{\Sigma}_{\theta} + \epsilon^2 \mathbf{I}) q(\theta'|\theta)} \right) \quad (12)$$

We now derive a procedure to estimate the probability of making an error in an accept/reject decision ($\mathcal{E}(\alpha)$, the *Metropolis-Hastings error*) and a threshold τ for actually making the decision. The error of making an incorrect decision can either be measured conditioned on $u \sim \mathcal{U}(0, 1)$ (the uniformly distributed draw used in the MH decision), or unconditionally, by integrating over $\mathcal{U}(0, 1)$. First we start with the conditional error which trivially extends to the unconditional error by averaging.

If $u \leq \tau$, then we accept the MH proposal and move to the proposed state. The probability of making an error in this case is $P(\alpha < u)$ (i.e. the probability we should actually reject):

$$P(\alpha < u) = \frac{1}{M} \sum_m [\alpha^{(m)} < u] \quad (13)$$

Similarly, if $u > \tau$ then we reject, and the error is $P(\alpha > u)$ (i.e. the probability we should actually accept):

$$P(\alpha > u) = \frac{1}{M} \sum_m [\alpha^{(m)} \geq u] \quad (14)$$

The total conditional error is therefore:

$$\mathcal{E}_u(\alpha) = [u \leq \tau] P(\alpha < u) + [u > \tau] P(\alpha \geq u) \quad (15)$$

and the total unconditional error is:

$$\mathcal{E}(\alpha) = \int \mathcal{E}_u(\alpha) \mathcal{U}(0, 1) du \quad (16)$$

which can again be estimated by Monte Carlo or grid values of u . The analytic calculation of $\mathcal{E}(\alpha)$ is the area under the cumulative distribution function of $p(\alpha)$ folded at τ (see Figure 1). This integral is also known as the *mean absolute deviation* [30] which is minimized at the median of $p(\alpha)$ (the value of α where the CDF equals 1/2), justifying our decision threshold $\tau = \text{median}(\alpha)$ (also determined by samples $\alpha^{(m)}$).

With this in hand, we now have the necessary tools to construct an adaptive synthetic-likelihood MCMC algorithm that uses $\mathcal{E}(\alpha)$ as a guide for running simulations (Algorithm 2). At the start of each MH step, S_0 simulations are run for both θ and θ' ; estimators are computed; then M $\alpha^{(m)}$ are sampled. Based on these samples, the median and $\mathcal{E}(\alpha)$ is computed. Note that this phase of the algorithm is very cheap; here we are sampling from J bivariate Gaussian distributions to compute Monte Carlo estimates of τ and $\mathcal{E}(\alpha)$, so M can be set high without a computational hit, though in practice $M < 100$ should be fine. While $\mathcal{E}(\alpha) > \xi$, ΔS new simulations are run and the estimators updated, along with new draws of $\alpha^{(m)}$, etc. The user-defined error threshold ξ is a knob which controls both the accuracy and computational cost of the MCMC. New simulations can be run at either θ or θ' ; we run simulations at both locations, though selecting one over the other based on the higher individual mean uncertainty could result in fewer simulations. As S increases, the uncertainty around $p(\alpha)$ decreases such that $\mathcal{E}(\alpha) < \xi$; once this occurs, the MH is now *confident* and it proceeds using the usual acceptance test, with τ as the acceptance threshold.

In many cases, the actual number of simulations required at each step can be quite small, for example when one parameter setting is clearly better than another (where the median is at or close to 0 or 1). Nevertheless, there remains a serious drawback to this algorithm for expensive simulations: all simulations are discarded after each MH step; a great waste considering the result is a single binary decision. Using a Gaussian process surrogate, described in the next section, we will remember all simulations and use them to gradually improve the surrogate and as a consequence, eventually eliminate the need to run simulations.

4 Gaussian Process Surrogate ABC

As mentioned in the introduction, in many scientific disciplines simulations can be extremely expensive. The algorithms up till now all have the downside that at each MCMC update a minimum number of simulations needs to be conducted. This seems unavoidable unless we store the information of previous simulations and use them to make accept/reject decisions in the future. In particular, we can *learn* the mean and covariance $\mu_{\theta}, \Sigma_{\theta}$ of the synthetic likelihood as a function of θ and as such avoid hav-

Algorithm 2 Adaptive Synthetic-likelihood ABC MH step

inputs: $q, \theta, \pi(\mathbf{x}|\theta), S_0, \Delta S, \epsilon, \mathbf{y}, \xi$
 $\theta' \sim q(\theta'|\theta)$
Init $c = 1, S = S_0$
repeat
 for $s = c : c + S$ **do**
 $\mathbf{x}'^{(s)} \overset{\text{sim}}{\sim} \pi(\mathbf{x}|\theta'), \mathbf{x}^{(s)} \overset{\text{sim}}{\sim} \pi(\mathbf{x}|\theta)$
 end for
 Update $c = S, S = S + \Delta S$
 Set $\hat{\mu}_{\theta'}, \hat{\Sigma}_{\theta'}, \hat{\mu}_{\theta}, \hat{\Sigma}_{\theta}$ using Eqns 5 and 6.
 for $m = 1 : M$ **do**
 Sample $\mu_{\theta'}^{(m)}, \mu_{\theta}^{(m)}$ using Eqn 11
 Set $\alpha^{(m)}$ using Eqn 12
 end for
 Set $\tau = \text{median}(\alpha^{(m)})$
 Set $\mathcal{E}(\alpha)$ using Eqn 16
until $\mathcal{E}(\alpha) < \xi$
if $\mathcal{U}(0, 1) \leq \tau$ **then**
 return θ'
end if
return θ

ing to perform simulations to compute them. There is a very natural tool that provides exactly this functionality, namely the Gaussian process (GP). For our purposes, the GP will “store” the simulation runs θ_n, \mathbf{x}_n for all simulations conducted during the MCMC run. We will use the GP as a “surrogate” function for the simulated statistics from which will be able to estimate the marginal likelihood values away from regions where actual simulations were run. Importantly, the GP provides us with uncertainty estimates of the marginal likelihood which will inform us of the need to conduct additional experiments in order to make confident accept/reject decisions. Going from the synthetic likelihood model to the GP represents a change from frequentist statistics in favor of (nonparametric) Bayesian statistics. Gaussian processes have recently also become a popular tool in the machine learning literature as surrogates of expensive regression surfaces, such as log-likelihoods [17]; optimization surfaces [8, 24]; simulations of physical systems [6]; emulation of computer codes [10]; and for accelerating ABC [28].

Similar in spirit to our own work, [28] uses GP surrogates to model the ABC log-likelihood surface in successive waves of inference, each eliminating regions of implausibility and producing more and more accurate models of the log-likelihood. There are two important differences. Space-filling design points are used by [28] to train their GP models, whereas we control the simulations with ξ , and we model all J simulation outputs versus a single log-likelihood, which is a much larger overhead for our approach, but has advantages, e.g. enabling posterior analysis and evaluation of the simulator.

Our surrogate model and algorithm follow directly from the synthetic-ABC approximation and randomized acceptance algorithm. The main difference between the two is that in this paper, we model the J statistics as J independent Gaussian processes (recall J is the dimensionality of \mathbf{y}). We note that it would be better to model the J statistics using a single joint Gaussian process. This can be done using “co-Kriging” or variants thereof [12, 7]. Although the independence assumption may lead to overconfidence (because it is assuming—falsely—independent evidence), it is also more robust in high-dimensions where the estimator of the full output covariance has high variance (it overfits). It may be that the mentioned multi-output GPs can provide an appropriate solution by tying the covariance structure across parameter space using a small number of kernel hyperparameters. For our experiments we found that independent GPs worked well enough to illustrate the algorithm’s potential. For high-dimensional outputs, modeling the log-likelihood directly may be more appropriate [28].

For each statistic j , the surrogate provides the following conditional predictive distribution of the expected value of statistic j :

$$\mu_{\theta j} \sim \mathcal{N}(\bar{\mu}_{\theta j}, \sigma_{\theta j}^2) \quad (17)$$

where the mean and covariance are determined by the set of N training inputs $\{\theta_n\}$ and N training outputs $\{\mathbf{x}_n\}$ (using only statistic j). They are given by the usual expressions for the GP mean and covariance,

$$\bar{\mu}_{\theta j} = \mathbf{k}_{\theta \Theta j} [\mathbf{K}_{\Theta \Theta j} + \sigma_j^2 \mathbf{I}]^{-1} \mathbf{X}[:, j] \quad (18)$$

and

$$\sigma_{\theta j}^2 = k_{\theta \theta j} - \mathbf{k}_{\theta \Theta j} [\mathbf{K}_{\Theta \Theta j} + \sigma_j^2 \mathbf{I}]^{-1} \mathbf{k}_{\Theta \theta j}$$

where $\mathbf{k}_{\theta \Theta j}$ is a 1 by N vector of kernel evaluations for the j ’th Gaussian process between θ and the input training set Θ , $\mathbf{K}_{\Theta \Theta j}$ is the j th kernel matrix evaluated on the training data; σ_j^2 is the data noise term for the j ’th statistic (used below in the acceptance ratio), \mathbf{X} is the N by J training output data set and $\mathbf{X}[:, j]$ is column j from the training data, and $k_{\theta \theta j}$ is a single kernel evaluation at θ for Gaussian process j .

The GPS-ABC algorithm is now run as follows (Algorithm 3). At each MH step, using Eqn 17, and for each j , M independent draws of $\mu_{\theta'}$ and μ_{θ} are sampled from their conditional predictive distribution. Note that this significantly different from the SL-ABC because there are now no default simulations to be run at each MH step; instead, the current surrogate model is used to predict both the expectation and uncertainty in simulation output. As before, Monte Carlo statistics are computed from acceptance probabilities $\alpha^{(m)}$ as follows,

$$\alpha^{(m)} = \min \left(1, \frac{\pi(\theta') \prod_j \mathcal{N}(y_j | \mu_{\theta' j}^{(m)}, \sigma_j^2 + \epsilon^2) q(\theta | \theta')}{\pi(\theta) \prod_j \mathcal{N}(y_j | \mu_{\theta j}^{(m)}, \sigma_j^2 + \epsilon^2) q(\theta' | \theta)} \right) \quad (19)$$

Algorithm 3 GPS-ABC MH step

inputs: $q, \theta, \pi(\mathbf{x}|\theta), S_0, \Delta S, \epsilon, \mathbf{y}, \xi$
 $\theta' \sim q(\theta'|\theta)$
repeat
 for $m = 1 : M$ **do**
 Sample $\mu_{\theta'j}^{(m)}, \mu_{\theta j}^{(m)}$ using Eqn 17
 Set $\alpha^{(m)}$ using Eqn 19
 end for
 Set $\tau = \text{median}(\alpha^{(m)})$
 Set $\mathcal{E}(\alpha)$ using Eqn 16
 if $\mathcal{E}(\alpha) > \xi$ **then**
 Acquire new training point.
 end if
until $\mathcal{E}(\alpha) < \xi$
if $\mathcal{U}(0, 1) \leq \tau$ **then**
 return θ'
end if
return θ

The error $\mathcal{E}(\alpha)$ and acceptance threshold τ are computed from the M samples; if $\mathcal{E}(\alpha) > \xi$, then a procedure for acquiring training points (i.e. simulations) is run, with the objective of reducing uncertainty for this specific MH step. Again, as with the adaptive synthetic likelihood algorithm, computing the M samples is very cheap. The procedure is then free to select any input location to run a simulation (whereas before we were forced to run at either θ or θ'), though the new simulation should be impactful for the current MH step. This means that we can choose locations other than θ and θ' , perhaps trying to limit the number of future simulation runs required in the vicinity. Analogous to acquisition functions for Bayesian optimization [8], actively acquiring points has the implicit goals of speeding up MCMC, reducing MCMC error, and limiting simulations. We have intentionally left vague the procedure for acquiring training points; for now we run simulations at θ or θ' , even though this is an inefficient use of our surrogate. Once a new training point is selected and run, the training input-output pair is added to all J Gaussian processes and the model hyperparameters may or may not be modified (with a small number of optimization steps or by sampling).

The key advantage of GPS-ABC is that with increasing frequency, we will not have to do any expensive simulations whatsoever during a MH step because the GP surrogate is sufficiently confident about the statistics' surface in that region of parameter space.

4.1 Theoretical Aspects of GPS-ABC

Two of the main contributions of GPS-ABC are *MCMC under uncertainty* and the introduction of *memory* into the Markov chain; we consider these steps as the only way to reduce the number of expensive simulations and as such

a necessary ingredient to GPS-ABC. Nevertheless, they present major differences from typical Bayesian inference procedures.

We now address two major theoretical aspects of the GPS-ABC algorithm: the *approximate* and *adaptive* nature of GPS-ABC. Although we have postponed formal proofs for future work we have outlined their main arguments below.

GPS-ABC is *approximate* because at each MH-step there is some probability that the chain will make an error, and that this corresponds to an error in the stationary distribution of the Markov chain (i.e. it is an approximation to the stationary distribution). In [14], another approximate MCMC algorithm is presented and it provides a proof for an upper bound on the error in the stationary distribution. The main argument is that if the MH-step error is small and bounded (along with a few other mild conditions), then the error in stationary distribution is bounded as well. We feel GPS-ABC fits into this same proof framework.

GPS-ABC is also *adaptive* since the approximation to the stationary distribution changes as more training points are added to the Gaussian process (we are learning the surrogate as we run the MCMC). Two of the major requirements for a valid adaptive MCMC algorithm are *diminishing adaptation* and *ergodicity* [19]. GPS-ABC satisfies the former as the number of training points acquired over an MCMC run rapidly decreases over time. When the adaptation slows and becomes insignificant, the Markov chain resembles the algorithm in [14], which, as we stated above, provides a proof of bounded convergence to the stationary distribution (and hence ergodicity); therefore we believe that GPS-ABC satisfies the latter requirement.

5 Experiments

The main goal of our experiments is to show the correctness and computational gains of GPS-ABC. Our results indicate that correct posterior samples can be obtained by GPS-ABC with orders of magnitude fewer simulation calls than traditional ABC sampling procedures.

We perform experiments on two simulation problems: 1) a toy Bayesian inference problem (the exponential problem) and 2) inference of parameters in a chaotic ecological system (the blowfly problem). In the former, the true posterior distribution is known and therefore provides a useful test-case for determining the correctness and convergence properties of ABC algorithms. There is no ground truth in the latter problem, making inference much more difficult, but we can nevertheless assess it by the quality of raw simulation outputs and convergence to a set of chosen statistics.

Here is a brief description of the algorithms used in our experiments. We ran ϵ -tube rejection sampling (REJ); synthetic-likelihood (SL, both marginal and pseudo-marginal); and two adaptive ξ -ABC algorithms: adaptive

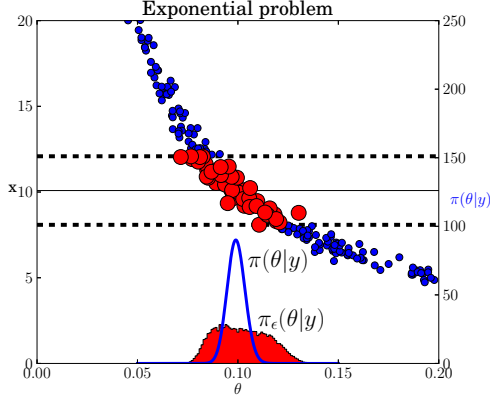


Figure 2: The exponential problem. Shown on the same y-axis are values from the simulator x and the density of the true and approximate posteriors ($\pi(\theta|y)$ and $\pi_\epsilon(\theta|y)$, respectively). The horizontal lines indicate $y \pm \epsilon$. The approximate posterior was generated from ϵ -tube rejection sampling ($\epsilon = 2$).

synthetic likelihood (ASL) and Gaussian process surrogate ABC (GPS). Rejection sampling is a procedure where θ is repeatedly drawn from the prior until all simulation statistics are within the ϵ -tube. This is repeated independently for each sample. Algorithms SL, ASL perform at least S simulations at each MH-step of a MCMC run; for marginal algorithms, this is $2S$ (the simulations at the current location are re-run), for ASL this can be higher, depending on ξ and due to randomness within the MH-step. Marginal versus pseudo-marginal results are indicated by the prefixes ‘m’ or ‘p’, respectively. Our MCMC algorithms are initialized with a single rejection sample, so initial simulation counts are affected by its ϵ value. GPS uses a small initial training set of size $S_0 = 50$. For the exponential problem, this training set was generated by rejection sampling. For the blowfly problem, a short MCMC run using SL was used to generate higher quality training points. GPS adapts its Gaussian process hyperparameters using MAP estimates found by conjugate gradient methods during initialization, and subsequently when the number of training points doubles (i.e. 100, 200, 400, etc). Finally, ASL and GPS adapt the number of simulations at each MH-step $\Delta S = 5$. Due to space constraints, some of the results are not shown in the main paper, but can be found in the supplementary material.

5.1 Inferring the parameter of an exponential distribution

In this illustrative problem we infer the rate of an exponential distribution under a gamma prior with shape α and rate β , having N observations at the true rate θ^* ; this is the *exponential example* in [27]. Let \mathbf{w} be a vector of N draws from an exponential distribution, i.e. $w_n \sim \text{Exp}(\theta)$. The posterior is a gamma distribution with shape $\alpha + N$ and rate $\beta + \sum w_n$. To use this problem with ABC, we use the exponential draws as the simulator and the mean of \mathbf{w} as the statistic y and assume that N is known. The inference

problem is therefore to sample from $p(\theta|y, \alpha, \beta, N)$.

For all runs we fixed $\alpha = \beta = 0.1$; a very broad prior over θ . Using the same random seed and $\theta^* = 0.1$, we generated $y = 10.0867$, which induces the θ -MAP value 0.09916 (not quite 10 and 0.1 due to their random draw and to a small influence from the prior). An illustration of this problem is shown in Figure 2.

Figure 3 show the results of running REJ, SL, and GPS to generate 50000 samples; this is repeated 10 times per algorithm. These three algorithms were chosen because they give roughly the same final error in distribution. In Figure 3a, the convergence to the (known) target posterior distribution (error), *per sample*, is shown. Figure 3b shows the same convergence in error, but is overlaid with the convergence *per simulation call* instead of per sample. Rejection sampling, as expected, provides the best convergence per sample (each sample is an independent sample from the approximate posterior) but computationally performs very poorly when ϵ is set to a value that gives small error in distribution.

As GPS-ABC acquires training points it is also sampling from the approximate posterior. Once the surrogate has learned the statistic surfaces in a region of parameter space, it no longer makes any simulation calls in that region. Eventually, the surrogate learns the surface for all the regions of parameter space where the posterior density is high. For this problem (with $\xi = 0.2$), this occurs after approximately 1000 simulations. As the MCMC run progresses, GPS-ABC gathers samples with decreasing amounts of computation. Figure 3c shows how the Gaussian process adaptation levels off during MCMC runs, whereas traditional ABC algorithms require a constant number of simulation calls per sample. As ξ decreases, more simulation calls are required for the GPS-ABC to model the statistics surfaces with increased precision; they all, however, have adaptation curves similar to Figure 3c.

5.2 Chaotic Ecological Systems

Adult blowfly populations exhibit dynamic behavior for which several competing population models exist. In this experiment, we use observational data and a simulation model from Wood [29], based on their improvement upon previous population dynamics theory. Population dynamics are modeled using (discretized) differential equations that can produce chaotic behavior for some parameter settings. An example blowfly replicate series is shown in Figure 4a, along with times-series generated by a sample from $\pi(\theta|y)$ using GPS-ABC.

In [29] there are several explanations of the population dynamics, corresponding to different simulations and parameters. We concentrate on the equation (1) in section 1.2.3 of the supplementary information, considered “a better al-

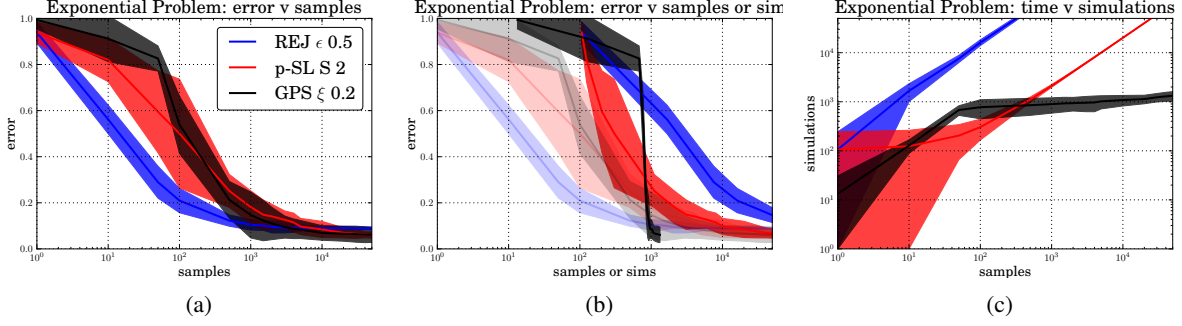


Figure 3: Convergence to target distribution for three algorithms: REJ ($\epsilon = 0.5$), p-SL ($S = 2$), and GPS ($\xi = 0.2$). The three have roughly the same final error in distribution. In (a) the convergence to the target *per sample* is shown; in (b) this convergence is overlaid with convergence *per simulation*. Around sample 1000, GPS has essentially stopped adapting, has learned the statistic surfaces, and no longer requires simulations. In (c) the growth of simulations per time step (i.e. a sample) for the sample algorithms is shown; both REJ and SL use a fixed number of simulations per iteration, whereas GPS stops adding new training points around time 1000. Note that we start the plot at the 10th sample, which requires a different number of simulations, depending on the algorithm.

ternative model” by the author. The population dynamics equation generates N_1, \dots, N_T using the following update rule:

$$N_{t+1} = PN_{t-\tau} \exp(-N_{t-\tau}/N_0)e_t + N_t \exp(-\delta\epsilon_t)$$

where $e_t \sim \mathcal{G}(1/\sigma_p^2, 1/\sigma_p^2)$ and $\epsilon_t \sim \mathcal{G}(1/\sigma_d^2, 1/\sigma_d^2)$ are sources of noise, and τ is an integer (not to be confused with the τ used as the MH acceptance threshold in our algorithms). In total, there are 6 parameters $\theta = \{\log P, \log \delta, \log N_0, \log \sigma_d, \log \sigma_p, \tau\}$. See [29] for further details about the significance of the parameters. We put Gaussian priors over all the parameters (with Gaussian proposal distributions), except for τ which has a Poisson prior (and a left/right increment proposal). Time-series generated with parameters from this prior distribution produce extremely varied results, some are chaotic, some are degenerate, etc. Modeling this simulator is *very* challenging.

As with any ABC problem the choice of statistics is important as it directly affects the quality of the results. It is also non-trivial and requires careful thought and sometimes trial and error. In total there are 10 statistics: the log of the mean of all 25% quantiles of $N/1000$ (4 statistics), the mean of the 25% quantiles of the first-order differences of $N/1000$ (4 statistics), and the maximal peaks of smoothed N , with 2 different thresholds (2 statistics). With these statistics it is possible to reproduce time-series that appear similar to the observations. Note that these statistics are different from Wood’s, but they capture similar time-series features and are sufficient to produce credible population dynamics.

The first set of experiments for the blowfly problem is shown in Figure 4b. For these experiments we compared REJ ($\epsilon = 5$), corresponding to an acceptance rate of roughly 2%, p-SL ($S = 10$), and finally GPS ($\xi = 0.3$). For SL, a small $\epsilon = 0.5$ was required to improve mixing. This helped all algorithms, but was less important for GPS, though it can be important if the Gaussian processes are not properly calibrated. Each algorithm was

run 5 times collecting 10K samples each. The GPS model for this problem consists of $J = 10$ independent Gaussian processes with $D = 6$ inputs each. The GPS was initialized with $S_0 = 50$ samples from a short SL-ABC MCMC run. In Figure 4b we show the posterior distributions for $\log P$, $\log \delta$, and $\log N_0$. Rejection sampling produces much broader posteriors than both SL and GPS, though they all share roughly the same mode. Between SL and GPS there is little difference in mode or shape, though GPS appears to have tighter confidence intervals. The real difference is the computational effort required: GPS used only 384 simulations to produce 10K, roughly 0.04 simulations per sample, whereas SL and REJ require 10 and 45 simulations for a single sample, respectively. Of course, GPS has much higher efficiency in practice, as the value 0.04 simulations per sample decreases over time and eventually reaches 0.

The second set of experiments examined the convergence properties of GPS compared to the other algorithms, focusing on the quality of the posterior predictive distributions *per sample* versus *simulation call*. For the blowfly data we do not have the ground-truth θ^* , but we do have the statistics of the observations for which we can monitor convergence. We do this by evaluating the posterior predictive distribution $p(\mathbf{y}|\mathbf{y}^*)$. Please note that we slightly change notation when discussing posterior predictive distributions by denoting \mathbf{y}^* the observed statistics, and \mathbf{y} as statistics generated by the posterior samples $p(\theta|\mathbf{y}^*)$. Pairs \mathbf{y} and θ are generated, with the exception of GPS, during the MCMC run, and are exactly the quantities we require for $p(\mathbf{y}|\mathbf{y}^*)$. Convergence will tell us the amount of computational effort required for an independent and (un)biased sample.

In Figure 4c we show the convergence in expected value of \mathbf{y} to \mathbf{y}^* per simulation. This is calculated in an online fashion by averaging statistics generated at θ (each *pos-*

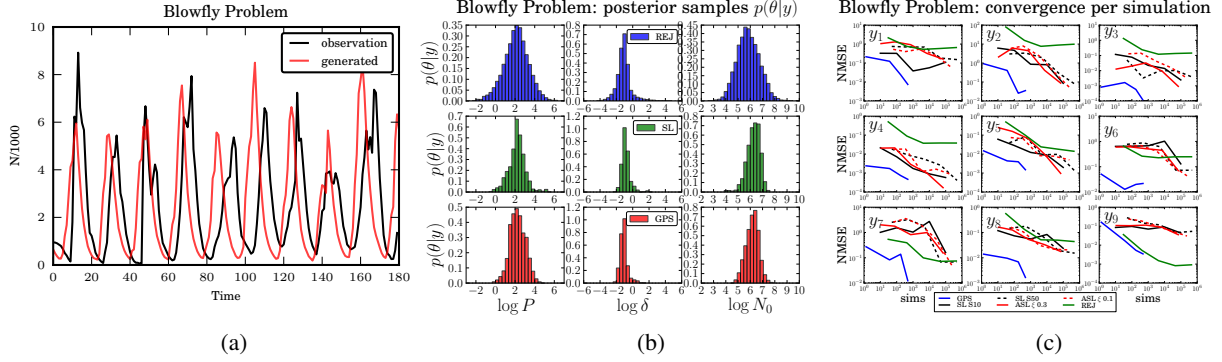


Figure 4: **(a)** The blowfly population time-series. The observations are in black lines and a generated time-series is shown in red. Note that θ is a sample from the GPS posterior ($\xi = 0.3$). See paper for description of the statistics. **(b)** Blowfly $p(\theta|y)$ for different algorithms. The top (blue) row is REJ $\epsilon = 5$, the middle (green) is pseudo-marginal SL S10, and the bottom (red) is GPS $\xi = 0.3$. Their respective simulations per sample ratios were: 45, 10, and 0.04. I.e. REJ had an acceptance rate of 2.2%, and GPS only used 384 simulations for 10K samples. Similar distributions were observed for ASL and other pseudo-marginal SL. **(c)** Convergence to y^* for different algorithms, using normalized mean-squared error. Each sub-plot shows the convergence to a single statistic as a function of simulation calls.

terior θ sample is allowed one y). We show the log-log plots of the normalized mean squared error (NMSE) versus number of simulations for the first 9 statistics. Algorithms run in this experiment were GPS ($\xi = 0.2$), p-SL ($S = 10, 50$), ASL ($\xi = 0.3, 0.1$), and REJ ($\epsilon = 5$). GPS not only converges systematically to y^* but does so with dramatically less effort. For some statistics REJ performed reasonably well, but in general exhibited significant bias. Both SL and ASL converged to similar biases, usually outperforming REJ, but in some cases was worse. Parameter settings where more computation was required for SL and ASL did result in slightly improved convergence, but the gain comes with a significantly higher computational cost. In summary, GPS is able to model the statistic surfaces, enabling it to correctly sample from the approximate posterior target distribution with higher precision and with orders of magnitude fewer simulation calls than the other algorithms.

6 Discussion and Future Work

We have presented a promising framework for performing inference in expensive, simulation-based models. Our algorithms improve current state-of-the-art ABC methods in that they require many fewer calls to the simulator, sometimes orders of magnitude fewer.

Using GPs for surrogate modeling has an appealing elegance; as nonparametric Bayesian models, they naturally incorporate both model and pseudo-data uncertainty into the MCMC algorithms. However, there are several technical issues and modeling limitations with GPs used for surrogate modeling. Heteroskedastic noise is more likely the norm than the exception for complicated simulators. The blowfly simulator is a prime example of this. Improvements to our GPs may be achieved using an input-dependent noise model [11, 13], where the noise is an additional independent Gaussian process. Another limitation

of our GP model is the output independence assumption. A more realistic assumption is a full covariance Gaussian process such as the convolution processes of [12, 7, 1]. One final limitation is GP calibration. We found that initializing the Gaussian process with rejection samples produced inferior results, as it tended to adapt its hyper-parameters optimally for those training points and had difficulty readapting. Despite these limitations, we feel that GPS-ABC deserves a place within the ABC toolbox.

Our GPS-ABC uses a random-walk proposal distribution which is inefficient for exploring the target distribution. Using GPs offers the opportunity to use other techniques to improve the mixing (and in turn computational cost). For example, in [17] a Hamiltonian Monte Carlo run on the GP surface is used to generate *independent* proposals. If applied to ABC, their algorithm would require the equivalent of a full simulation at the proposed location, whereas if we incorporated a similar technique, we would then test the GP uncertainty to determine if a simulation was required.

There has been a recent surge in interest in Bayesian optimization using Gaussian process (GP-BO) surrogates of the objective surface [8, 24]. GP-BO is often applied to problems where simulation or sometimes user feedback guides the surrogate’s construction. What is most interesting about GP-BO is its use of model uncertainty to *actively* determine the next simulation location implemented through acquisition functions. These ideas can be generalized to our GPS-ABC algorithm to further reduce simulation costs, while at the same time maintaining control of the MCMC error.

Acknowledgements

MW acknowledges support from the CIFAR-NCAP program.

References

- [1] Álvarez, M. and Lawrence, L.D. Computationally efficient convolved multiple output gaussian processes. *Journal of Machine Learning Research*, 12:1425–1466, 2011.
- [2] Andrieu, C. and Roberts, G. The pseudo-marginal approach for efficient monte carlo computations. *The Annals of Statistics*, 37(2):697–725, 2009.
- [3] Bazin, Eric, Dawson, Kevin J, and Beaumont, Mark A. Likelihood-free inference of population structure and local adaptation in a bayesian hierarchical model. *Genetics*, 185(2):587–602, 2010.
- [4] Beaumont, M.A., Cornuet, J.-M., Marin, J.-M., and Robert, C.P. Adaptive approximate bayesian computation. *Biometrika*, 96(4):983–990, 2009.
- [5] Beaumont, Mark A, Zhang, Wenyang, and Balding, David J. Approximate bayesian computation in population genetics. *Genetics*, 162(4):2025–2035, 2002.
- [6] Billionis, I., Zabarar, N., Konomi, B.A., and Lin, G. Multi-output separable gaussian process: Towards an efficient fully bayesian paradigm for uncertainty quantification. *Journal of Computational Physics*, 241:212–239, 2013.
- [7] Boyle, P. and Frean, M. Dependent gaussian processes. *Advances in Neural Information Processing Systems 17*, 2005.
- [8] Brochu, E., Cora, Vlad M., and de Freitas, Nando. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. Technical report, UBC Technical Report, 2010.
- [9] Ceperley, D.M. and Dewing, M. The penalty method for random walks with uncertain energies. *Journal of Chemical Physics*, 110(20):9812–9820, 1999.
- [10] Conti, S., Gosling, J.P., Oakley, J., and O’Hagan, A. Gaussian process emulation of dynamic computer codes. *Biometrika*, 96(3):663–676, 2009.
- [11] Goldberg, P.W., Williams, C.K.I., and Bishop, C.M. Regression with input-dependent noise: A gaussian process treatment. *Advances in Neural Information Processing Systems 10*, 1998.
- [12] Higdon, D. Space and space-time modeling using process convolutions. In Anderson, C.W., Barnett, V., Chatwin, P.C., and Abdel, E.-S.H. (eds.), *Quantitative Methods for Current Environmental Issues*, pp. 37–56. Springer London, 2002. ISBN 978-1-4471-1171-9.
- [13] Kersting, K., Plagemann, C., Pfaff, P., and Burgard, W. Most likely heteroscedastic gaussian process regression. In *Proceedings of the 24th International Conference on Machine Learning*, pp. 393–400. ACM, 2007.
- [14] Korattikara, A., Chen, Y., and Welling, M. Austerity in mcmc land: Cutting the metropolis-hastings budget. *ICML*, 2014. To appear.
- [15] Marin, J.-M., Pudlo, P., Robert, C.P., and Ryder, R.J. Approximate bayesian computational methods. *Statistics and Computing*, 22:1167–1180, 2012.
- [16] Nicholls, G.K., Fox, C., and Watt, A.M. Coupled mcmc with a randomized acceptance probability. Technical report, arXiv:1205.6857v1, 2012.
- [17] Rasmussen, C.E. Gaussian processes to speed up hybrid monte carlo for expensive bayesian integrals. *Bayesian Statistics*, 7:651–659, 2003.
- [18] Ratmann, O., Jorgensen, O., Hinkley, T., Stumpf, M., Richardson, S., and Wiuf, C. Using likelihood-free inference to compare evolutionary dynamics of the protein networks of h. pylori and p. falciparum. *PLoS Computational Biology*, 3(11):e230, 2007.
- [19] Roberts, G.O. and Rosenthal, J.S. Coupling and ergodicity of adaptive mcmc. *J. Appl. Prob.*, 44:458–475, 2007.
- [20] Schafer, C.M. and Freeman, P.E. Likelihood-free inference in cosmology: Potential for the estimation of luminosity functions. In *Statistical Challenges in Modern Astronomy V*, pp. 3–19. Springer, 2012.
- [21] Sisson, SA, Fan, Y, and Tanaka, Mark M. Sequential monte carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 104(6):1760, 2007.
- [22] Sisson, S.A., Peters, G.W., Briers, M., and Fan, Y. A note on target distribution ambiguity of likelihood-free samplers. *Arxiv preprint arXiv:1005.5201v1 [stat.CO]*, 2010.
- [23] Sisson, Scott A and Fan, Yanan. Likelihood-free markov chain monte carlo. *Arxiv preprint arXiv:1001.2058*, 2010.
- [24] Snoek, J., Larochelle, H., and Adams, R.P. Practical bayesian optimization of machine learning algorithms. *Advances in Neural Information Processing Systems 25*, 2012.
- [25] Tavaré, S., Balding, D.J., Griffiths, R.C., and Donnelly, P. Inferring coalescence times from dna sequence data. *Genetics*, 145(2):505–518, 1997.
- [26] Toni, T., Welch, D., Strelkowa, N., Ipsen, A., and Stumpf, M.P.H. Approximate bayesian computation scheme for parameter inference and model selection in dynamical systems. *Journal of the Royal Society Interface*, 6(31):187–202, 2009.
- [27] Turner, Brandon M and Van Zandt, Trisha. A tutorial on approximate bayesian computation. *Journal of Mathematical Psychology*, 56(2):69–85, 2012.
- [28] Wilkinson, R. Accelerating abc methods using gaussian processes. *AISTATS*, 2014.
- [29] Wood, Simon N. Statistical inference for noisy nonlinear ecological dynamic systems. *Nature*, 466(7310):1102–1104, 2010.
- [30] Xue, Jing-Hao and Titterton, D. Michael. The p -folded cumulative distribution function and the mean absolute deviation from the p -quantile. *Statistics and Probability Letters*, 81:1179–1182, 2011.

Lifted Message Passing as Reparametrization of Graphical Models

Martin Mladenov

TU Dortmund University
{fn.ln}@cs.tu-dortmund.de

Amir Globerson

Hebrew University
gamir@cs.huji.ac.il

Kristian Kersting

TU Dortmund University
{fn.ln}@cs.tu-dortmund.de

Abstract

Lifted inference approaches can considerably speed up probabilistic inference in Markov random fields (MRFs) with symmetries. Given evidence, they essentially form a lifted, i.e., reduced factor graph by grouping together indistinguishable variables and factors. Typically, however, lifted factor graphs are not amenable to off-the-shelf message passing (MP) approaches, and hence requires one to use either generic optimization tools, which would be slow for these problems, or design modified MP algorithms. Here, we demonstrate that the reliance on modified MP can be eliminated for the class of MP algorithms arising from MAP-LP relaxations of pairwise MRFs. Specifically, we show that a given MRF induces a whole family of MRFs of different sizes sharing essentially the same MAP-LP solution. In turn, we give an efficient algorithm to compute from them the smallest one that can be solved using off-the-shelf MP. This incurs no major overhead: the selected MRF is at most twice as large as the fully lifted factor graph. This has several implications for lifted inference. For instance, running MPLP results in the first convergent lifted MP approach for MAP-LP relaxations. Doing so can be faster than solving the MAP-LP using lifted linear programming. Most importantly, it suggests a novel view on lifted inference: it can be viewed as standard inference in a reparametrized model.

1 INTRODUCTION

Probabilistic logical languages [5] provide powerful formalisms for knowledge representation and inference. They allow one to compactly represent complex relational and uncertain knowledge. For instance, in the friends-and-smokers Markov logic network (MLN) [17], the weighted

formula 1.1 : $\text{fr}(X, Y) \Rightarrow (\text{sm}(X) \Leftrightarrow \text{sm}(Y))$ encodes that friends in a social network tend to have similar smoking habits. Yet, performing inference in these languages is extremely costly, especially if it is done at the propositional level. Instantiating all atoms from the formulae in such a model induces a standard graphical model (potentially) with symmetries, i.e., with repeated factor structures for all grounding combinations. Recent advances in lifted probabilistic inference [16] such as [3, 15, 1, 14, 18] (see [9] for an overview that also covers exact inference approaches), have rendered many of these large, previously intractable models quickly solvable by exploiting the induced symmetries. For instance, lifted message-passing (MP) approaches such as [19, 10, 22, 8, 1] have been proven successful in several important AI applications such as link prediction, social network analysis, satisfiability and boolean model counting problems. Lifted MP approaches such as lifted Belief Propagation (BP) first automatically group together variables and factors of the graphical model into supervariables and superfactors if they have identical computation trees (i.e., the tree-structured “unrolling” of the graphical model computations rooted at the nodes). Then, they run modified MP algorithms on this lifted network. These modified MP algorithms, however, can also be considered a downside of today’s lifted MP approaches. They require more information than is actually captured by a standard factor graph. More precisely, lifted MP will typically exponentiate a message from a supervariable to a superfactor by the count of ground instances of this superfactor, which are neighbors to a ground instance of the supervariable. Since these multi-dimensional counts have to be stored in the network, the lifted factor graph becomes a multigraph (i.e., a factor graph with edge counts and self-loops), in contrast to a standard factor graph where no multiedges or loops are allowed. Hence, lifted factor graphs are not amenable to off-the-shelf MP approaches. Instead, lifted MP has its own ecosystem of lifted data structures and lifted algorithms. In this ecosystem, considerable effort is required to keep up with the state of the art in propositional inference.

In this paper we demonstrate that the reliance on modified MP can be eliminated for the class of MP algorithms arising

ing from linear programming (LP) relaxations of MAP inference (MAP-LPs) of pairwise MRFs. MAP-LPs approximate the MAP problem as an LP with polynomially many constraints [25], which is therefore tractable, and have several nice properties. First, they yield an upper bound on the MAP value, and can thus be used within branch and bound methods. Second, they provide certificates of optimality, so that one knows if the problem has been solved exactly. Third, the LP can be solved using simple algorithms such as coordinate descent, many of which have a nice message passing structure. [12] Fourth, the LP relaxations can be progressively tightened by adding gradually constraints of a higher order. This has been shown to solve challenging MAP problems [20].

Indeed, it is already known that MAP-LP relaxations of MRFs can be lifted efficiently [13, 3, 14], and the resulting lifted LPs can be solved using any off-the-shelf LP solver¹. Unfortunately, however, the liftings employed there may not preserve the MRF structure of the underlying LP. That is, if we lift a MAP-LP, we end up with constraints that do not conform to the MAP-LP template as already observed by Bui et al. (see Section 7 in [3]). In turn, existing MP solvers for MAP-LPs such as MPLP and TRW-BP — that have been reported to often solve the MAP-LP significantly faster than generic LP solvers — will not work without modifying them. Doing so, however, takes a lot of effort (if it is at all possible): it has to be done for each existing MP approach separately; there is no general methodology for doing this, and the extra coding itself is error prone. Hence this “upgrading methodology” may significantly delay the development of lifted MP approaches. Fortunately, as we demonstrate here, the theory of lifted LPs provides us with a way around these issues. The main insight is that a given MRF induces actually a whole family of MRFs of different sizes sharing essentially the same MAP-LP solution. From these, one can select the smallest one where MAP beliefs can be computed using off-the-shelf MP approaches. These beliefs then are also valid (after a simple transformation) for the original problem. Moreover, this incurs no major overhead: the selected MRF is at most twice as large as the fully lifted factor graph. In this way we eliminate the need for modified MP algorithms.

To summarize, our contributions are two-fold. **(1)** By making use of lifted linear programming, we show that LP-based lifted inference in MRFs can be formulated as ground inference on a reparametrized MRF. **(2)** We give an efficient algorithm that given a ground MRF finds the smallest reparametrized MRF and show that its size is not more than twice the size of the fully lifted model.

¹A similar approach has been proposed for exact MAP by Noesner et al. [15]. Moreover, Sarkhel et al. [18] have recently shown that MAP over MLNs can be reduced to MAP over Markov networks if the MLN has very restrictive properties. In contrast our approach is generic for MAP-LP relaxations.

This has several implications for lifted inference. For instance, using MPLP [6] results in the first convergent MP approach for MAP-LP relaxations, and using other MP approaches such as TRW-BP [24] actually spans a whole family of lifted MP approaches. This suggests a novel view on lifted probabilistic inference: *it can be viewed as standard inference in a reparametrized model*.

We proceed as follows. We start off with reviewing MAP-LP basics. Then, we touch upon equitable partitions and lifted LPs, and use them to develop the reparametrization approach. Before concluding we provide empirical illustrations, which support our theoretical results.

2 BACKGROUND

We start off by introducing MAP inference and its LP relaxation. Then we will touch upon equitable partitions and recall how they can be used in lifted linear programming.

MAP Inference in MRFs. Let $\mathbf{X} = (X_1, X_2, \dots, X_n)$ be a set of n discrete-valued random variables and let x_i represent the possible realizations of random variable X_i . Markov random fields (MRFs) compactly represent a joint distribution over \mathbf{X} by assuming that it is obtained as a product of functions defined on small subsets of variables [11]. For simplicity, we will restrict our discussion to a specific subset of MRFs, namely Ising models with arbitrary topology². In an Ising model $I = (G, \theta)$ on a graph $G = (V, E)$, all variables are binary, i.e., $\mathbf{X}_i \in \{0, 1\}$. Moreover, in an Ising model G must be a *simple* graph, i.e. G must have no self-loops or multiple edges between vertices. The model is then given by: $p(x) \propto \exp \left[\sum_{ij \in E} \theta_{ij} x_i x_j + \sum_i \theta_i x_i \right]$. In the following we will find it convenient to represent Ising models as factor graphs. The factor graph of an Ising model combines the structure and parameters of the model into a single bipartite graph. In this graph we have a variable vertex v_i for each probabilistic variable \mathbf{X}_i and a factor vertex ϕ_i for each θ_i and ϕ_{ij} for θ_{ij} . Moreover, ϕ_i is connected to v_i and ϕ_{ij} to v_i and v_j . While for ground Ising models, the factor graph does not capture any additional information, it makes the presentation of lifted structures simpler and reveals the essence of the conflict between lifting and message-passing. Hence, from now on when we refer to an Ising model $I = (G, \theta)$, by G we will mean the corresponding factor graph.

The Maximum a-posteriori (MAP) inference problem is defined as finding an assignment maximizing $p(x)$. This can equivalently be formulated as the following LP

$$\mu^* = \arg\max_{\mu \in \mathcal{M}(G)} \sum_{ij \in E} \mu_{ij} \theta_{ij} + \sum_i \mu_i \theta_i = \theta \cdot \mu \quad (1)$$

²The shorter description of MAP-LP for Ising models makes the presentation easier. Our approach, however, can be applied to any pairwise MRF with only minor modifications.

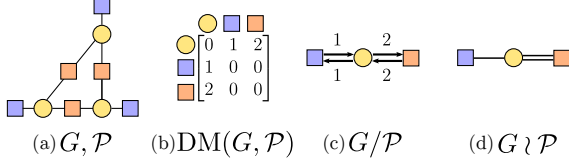


Figure 1: Lifted Structures: Example of a factor graph and its partitions and quotients. (a) A factor graph G (variables in circles, factors in squares) and its coarsest EP \mathcal{P} represented by the colors. (b) The degree DM matrix of G according to \mathcal{P} . (c) The corresponding quotient graph G/\mathcal{P} . (d) The factor quotient $G \wr \mathcal{P}$.

where the set $\mathcal{M}(G)$ is known as the marginal polytope [25]. Even though Eq. 1 is an LP, the polytope $\mathcal{M}(G)$ generally requires an exponential number of inequalities to describe [25], and is NP-complete to maximize over. Hence one typically considers tractable relaxations (outer bounds) of $\mathcal{M}(G)$. The outer bounds we consider are equivalent to the standard local consistency bounds typically considered in the literature (e.g., see [25] Eq. 8.32). However, we present them in a slightly different manner, which simplifies our presentation. Define the following set in $[0, 1]^{|V|+|E|}$:

$$L(G) = \left\{ \begin{array}{l} \mu \geq 0, \forall \phi_{ij} \in G : \\ \alpha(ij) \equiv \mu_{ij} \leq \mu_j, \beta(ij) \equiv \mu_{ij} \leq \mu_i; \\ \gamma(ij) \equiv \mu_i + \mu_j - \mu_{ij} \leq 1 \end{array} \right\}. \quad (2)$$

The polytope $L(G)$ is sometimes referred to as the local marginal polytope [21]. The vectors with $\{0, 1\}$ coordinates in $L(G)$ are the vertices of the $\mathcal{M}(G)$. In other words $\mathcal{M}(G)$ is the convex hull of $L(G) \cap \{0, 1\}^{|V|+|E|}$. We call the relaxed inference problem over $L(G)$ MAP-LP. Note that whenever $\mathcal{M}(G)$ and $L(G)$ do not coincide, $L(G)$ (which is an outer bound on $\mathcal{M}(G)$) has fractional vertices and the resulting LP may have optima which are not valid assignments. However, all integral points in $L(G)$ correspond to valid assignments, thus if the solution μ^* happens to be integral, then this μ^* solves the MAP problem.

Equitable Partitions (EPs) of Graphs and Matrices.

Lifted inference approaches essentially work with reduced models by grouping together indistinguishable variables and factors. In other words, they exploit symmetries. For linear programs, Mladenov *et al.* [13, 14] have shown that such symmetries can be formally captured by equitable partitions of weighted graphs and matrices. Since these partitions also play an important part in our argument we will next review the most relevant concepts and results.³ For an illustration, we refer to Fig. 1.

³Note, however, that the definitions we present here are tailored towards bipartite structures (e.g. factor graphs with variables and factors, matrices with rows and columns) for the sake of clarity. They are not the most general ones found in literature.

Let $U = V \cup F$ be a set consisting of two kinds of objects, e.g. the variables and factors of a factor graph as in Fig. 1(a), or the row and column indices of a matrix. A partition $\mathcal{P} = \{P_1, \dots, P_p\} \cup \{Q_1, \dots, Q_q\}$ is a family of disjoint subsets of U , such that $\bigcup_{i=1}^p P_i = V$ and $\bigcup_{i=1}^q Q_i = F$. In Fig. 1 the partition is indicated by the colors of the nodes. A convenient data structure for performing algebraic operations using partitions is the incidence matrix $B \in \{0, 1\}^{|U| \times |\mathcal{P}|}$. The incidence matrix shows the assignment of the elements of U to the classes of \mathcal{P} – it has one row for every object and one column for every class. The entry in the row of object u and the column of class P_p is

$$B_{up} = 1 \text{ if } u \in P_p \text{ and } 0 \text{ if } u \notin P_p.$$

We shall also make use of the normalized transpose of B , which we denote by $\hat{B} \in \mathbb{Q}^{|\mathcal{P}| \times |U|}$ and define as

$$\hat{B}_{pu} = 1/|P_p| \text{ if } u \in P_p \text{ and } 0 \text{ if } u \notin P_p.$$

Algebraically, B and \hat{B} are related as $\hat{B} = (B^T B)^{-1} B^T$, i.e., \hat{B} is the left pseudoinverse of B : $\hat{B} B = I_{|\mathcal{P}|}$.

The partitions we consider will never group elements of V with elements in F . Thus, the matrix B will always be of the form $B = \begin{pmatrix} B_P & 0 \\ 0 & B_Q \end{pmatrix}$, where B_P and B_Q correspond to the partitions of V and F respectively. We shall also use the notation $B = (B_P, B_Q)$ to refer to this block diagonal matrix, and similarly $\hat{B} = (\hat{B}_P, \hat{B}_Q)$.

Let $u \in \mathbb{R}^{|U|}$ be a real vector composed as $\mathbf{u} = [\mathbf{c}, \mathbf{b}]^T$, $\mathbf{c} \in \mathbb{R}^{|V|}$, $\mathbf{b} \in \mathbb{R}^{|F|}$. The values of \mathbf{u} can be thought of as labels for the elements of U . We say that a partition \mathcal{P} respects \mathbf{u} if for every $x, y \in U$ that are in the same class of \mathcal{P} , we have $u_x = u_y$. Note that if \mathcal{P} respects \mathbf{u} , then $(\mathbf{c}^T B_P)_i = |P_i| c_x$ where x is any member of P_i (and similarly for B_Q, \mathbf{b}). Moreover, $(\hat{B}_P \mathbf{c})_i = c_x$ where x is any member of P_i (and similarly for \hat{B}_Q, \mathbf{b}).

We next define a special class of partitions of graphs and matrices, which play a central role in our argument. Let us first consider a bipartite graph $G = (V \cup F, E)$. Here V and F are the two sides of the graph, and E are the edges connecting them. The neighbors of a node v in this graph are denoted by $\text{nb}(v)$.

Definition 1 (Equitable partition of a bipartite graph).

An equitable partition of a bipartite graph $G = (V \cup F, E)$ given a vector $\mathbf{u} \in \mathbb{R}^{|V|+|F|}$ is a partition $\mathcal{P} = \{P_1, \dots, P_p, Q_1, \dots, Q_q\}$ of the vertex set V and F such that (a) for every pair $v, v' \in V$ in some P_m , and for every class Q_n , $|\text{nb}(v) \cap Q_n| = |\text{nb}(v') \cap Q_n|$; (b) for every pair $f, f' \in F$ in some Q_m , and for every class P_n , $|\text{nb}(f) \cap P_n| = |\text{nb}(f') \cap P_n|$. Furthermore, \mathcal{P} must respect the vector \mathbf{u} .

If we are dealing with matrices, the above definition can be extended. Essentially, we view a matrix $A \in \mathbb{R}^{m \times n}$ as a weighted graph over the set $\{\text{row}[1], \dots, \text{row}[m]\} \cup$

$\{\text{col}[1], \dots, \text{col}[n]\}$, where A_{ij} is the weight of edge between $\text{row}[i]$ and $\text{col}[j]$. More precisely:

Definition 2 (Equitable partition of a matrix). An equitable partition of a matrix $A \in \mathbb{R}^{m \times n}$ given a vector \mathbf{u} is a partition $\mathcal{P} = \{P_1, \dots, P_p, Q_1, \dots, Q_q\}$ of the sets $V = \{\text{row}[1], \dots, \text{row}[m]\}$ and $F = \{\text{col}[1], \dots, \text{col}[n]\}$ s.t. **(a)** for every pair $v, v' \in V$ in some P_m , and for every class Q_n , $\sum_{f \in Q_n} A_{vf} = \sum_{f \in Q_n} A_{v'f}$ and **(b)** for every pair $f, f' \in F$ in some Q_m , and for every class P_n , $\sum_{v \in P_n} A_{vf} = \sum_{v \in P_n} A_{vf'}$. In addition, \mathcal{P} must respect \mathbf{u} .

Note that Def. 1 is an instance of Def. 2 when we take as A the (biparte) adjacency matrix of a graph G . An illustration of an equitable partition of a graph is given Fig. 1(a).

One notable kind of equitable partitions (EPs) are orbit partitions (OPs) – the partitions that arise under the action of the automorphism group of a graph or matrix. Their role in MAP inference has been studied in [3]. Although OP-based lifting is indeed practical in a number of cases or even the only applicable one, in particular for exact inference approaches, computing them is a GI-complete problem. Because of this we will stick to EPs which are more efficiently computable and yield more reduction (to be discussed shortly). Still, we would like to stress that our result applies to any EP, in particular to OPs.

Using an EP of a graph or a matrix, we can derive condensed representations of that graph or matrix using the partition. This is the essence of lifting: *the reduced representation is as good as the original representation for some computational task at hand, while (potentially) having a significantly smaller size*. A key insight that we exploit here is that there is a one-to-one relationship between EPs of the factor graph of an Ising model (as in Def. 1) and the EPs of its MAP-LP matrix (as in Def. 2).

One useful representation of a graph and its equitable partition is via a degree matrix, as illustrated in Fig. 1(b). The degree matrix, $\text{DM}(G, \mathcal{P})$, has $|\mathcal{P}| \times |\mathcal{P}|$ entries, where each entry represents how members of different classes interact. More precisely, $\text{DM}(G, \mathcal{P})_{ij} = |\text{nb}(u) \cap P_j|$, where u is any element of P_i . Due to the bipartiteness of G , this matrix will have the block form $\text{DM}(G, \mathcal{P}) = \begin{pmatrix} 0 & \text{DV} \\ \text{DF} & 0 \end{pmatrix}$, where DV represents the relationship of the P -classes to the Q -classes and DF vice-versa. As a shorthand, we use the notation $\text{DM} = (\text{DF}, \text{DV})$. Graphically (see Fig. 1(c)), a degree matrix can be visualized as a quotient graph G/\mathcal{P} , which is a directed multi-graph. In G/\mathcal{P} there is a node for every class of \mathcal{P} . Given two nodes u, v we have $|\text{nb}(u) \cap P_j|, u \in P_i$ many edges going from u to v . $\text{DM}(G, \mathcal{P})$ is essentially the weighted adjacency matrix of G/\mathcal{P} .

Later on, we will be interested in the interaction of the factors with variables rather than the other way around. Therefore, we introduce the factor quotient graph $G \wr \mathcal{P}$ of

G , which corresponds only to the DF-block of $\text{DM}(G, \mathcal{P})$ as shown in Fig. 1(d). That is, we draw only edges going from factor classes to variable classes, but not the other way around. Moreover, as our MRFs are pairwise, a factor class can have a degree of at most two to any variable class. We will thus not write numbers on top of the arcs, but draw double or single edges. To stay consistent with existing terminology, we call the nodes of $G \wr \mathcal{P}$ corresponding to variable classes of G “supervariables”, and factor-class nodes “superfactors”. Note that if \mathcal{P} is the OP of G , the resulting factor quotient is the “lifted graph” considered in [3].

Finally, to compute EPs one can use color-passing (also known as “color refinement” or “1-dimensional Weisfeiler-Lehman”). It is a basic algorithmic routine for graph isomorphism testing. It iteratively partitions, or colors, vertices of a graph according to an iterated degree sequence in the following fashion: initially, all vertices get their label in G as color, and then at each step two vertices that so far have the same color get different colors if for some color c they have a different number of neighbors of color c . The iteration stops if the partition remains unchanged. For matrices, a suitable extension was introduced in [7]. The resulting partition is called the *coarsest equitable partition* (CEP) of the graph, and can be computed asynchronously in quasi-linear time $O((n+m) \log n)$ (e.g., see [2]).

Lifted Linear Programming and Lifted MAP-LPs. By computing an EP of the matrix of a linear program (LP) one can derive a smaller but equivalent LP – the “lifted” LP – whose optimal solutions can easily be translated back to a solution of the original LP [13, 7]. This will be key in what follows, and is reviewed below.

Let $L = (A, \mathbf{b}, \mathbf{c})$ be a linear program, corresponding to the optimization problem $\mathbf{x}^* = \arg\max_{A\mathbf{x} \leq \mathbf{b}} \mathbf{c}^T \mathbf{x}$.

Theorem 3 (Lifted Linear Programs [7]⁴). Let $\mathcal{P} = \{P_1, \dots, P_p\} \cup \{Q_1, \dots, Q_q\}$ be an equitable partition with incidence matrix $B = (B_P, B_Q)$ of the rows and columns of A , which respects the vector $\mathbf{u} = [\mathbf{c}, \mathbf{b}]^T$. Then, $L' = (\hat{B}_Q A B_P, \hat{B}_Q \mathbf{b}, B_P^T \mathbf{c})$ is an LP with fewer variables and constraints. The following relates L and L' :

- (a) If \mathbf{x}' is a feasible point in L' , then $B_P \mathbf{x}'$ is feasible in L . If in addition \mathbf{x}' is optimal in L' , $B_P \mathbf{x}'$ is optimal in L with the same objective value.
- (b) if \mathbf{x} is a feasible point in L , then $\hat{B}_P \mathbf{x}$ is feasible in L' . If in addition \mathbf{x} is optimal in L , $\hat{B}_P \mathbf{x}$ is optimal in L' with the same objective value.

In previous works, only part (a) has been exploited. That is, as illustrated in Fig. 2, given any LP we construct L' using equitable partitions, solve it (often faster than the original one), and finally transfer the solution to the larger problem by virtue of (a) above. This “standard” way applies to MAP-LPs as follows (see [13, 7] for more details).

⁴We restrict the theorem to what is relevant for the discussion.

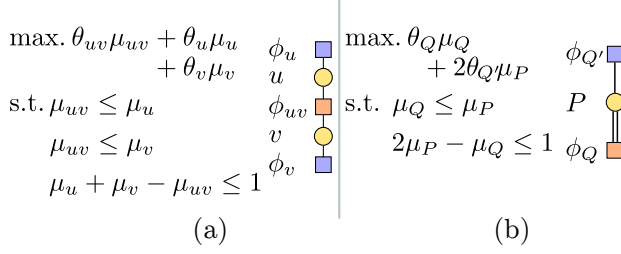


Figure 2: Illustration of lifting MAP-LPs. (a) A factor graph and the corresponding MAP-LP; (b) A factor quotient and the corresponding lifted MAP-LP. This example also illustrates that the lifted MAP-LP in (b) is not amenable to standard MP anymore, since the constraint $2\mu_P - \mu_Q \leq 1$ does not appear in standard MAP-LPs. However, note that the lifted structure is identical to the one in Fig. 1(d), yet the original factor graph in (a) above is smaller. That is, factor graphs of different sizes may share the (structurally) same lifted MAP-LP.

Given an MRF with graph G and parameter θ , which we denote by $I = (G, \theta)$, denote its standard MAP-LP relaxation by the LP defined via $(A, \mathbf{b}, \mathbf{c})$. To obtain a potentially smaller LP, we calculate the equitable partition of the LP, and its corresponding B, \hat{B} matrices. The LP defined via $(\hat{B}_Q A B_P, \hat{B}_Q \mathbf{b}, B_P^T \mathbf{c})$ is then equivalent to the original MAP-LP in the sense of Thm. 3. We refer to this LP as LMAP-LP(I).

Unfortunately, as we will show in the next section, LMAP-LP(I) is not a standard MAP-LP. In turn, it is not amenable to standard message passing based MAP-LP solvers. Fortunately, by making heavily use of part (b) of Thm. 3 as well, we will show how to produce LPs that have this special structure. This results “lifting by reparametrization”.

3 LIFTING BY REPARAMETRIZATION

To introduce the “lifting by reparametrization” approach we proceed in two steps. First, we introduce the class of all MRFs whose MAP-LPs are equivalent, in the sense that solving one such LP results in a solution to all LPs in the class. Then, we will show how to construct the smallest such equivalent MRFs for a given MRF in the class.

LP-equivalence of MRFs. We start off by discussing the structure of the lifted MAP-LP (LMAP-LP). The main purpose is to illustrate why lifting generally does not preserve the message-passing structure of the LP (see also Fig. 2). Then, as an alternative, we introduce an equivalence theorem in the spirit of Thm. 3. As illustrated in Fig. 3, instead of relating ground (original) and lifted LPs, it relates ground LPs of different sizes that have the same lifting. This is the main insight underlying our “lifting by reparametrization” approach: *instead of lifting the ground MAP-LP of an MRF at hand, we replace it by the ground*

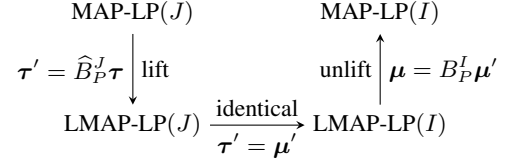


Figure 3: Commutative diagram established by Thm. 6 underlying our reparametrization approach to lifted inference.

LP of another equivalent MRF, hopefully of much smaller size. In the following we will now formally define what “equivalence” means here.

Our starting point is to note that LMAP-LP(I) only depends on I via the structure of $G \wr \mathcal{P}$. To provide the formal result, we need a few more notations. First, the graph $G \wr \mathcal{P}$ has nodes corresponding to groups of variables in the original factor graph, and nodes corresponding to groups of factors in the original factor graph. We denote those by $V(G \wr \mathcal{P})$ and $F(G \wr \mathcal{P})$ respectively. Second, $G \wr \mathcal{P}$ is a multigraph and may have multiple edges between its nodes. We define $\text{nb}(Q)$ to be the neighbors of $Q \in F(G \wr \mathcal{P})$ in $G \wr \mathcal{P}$, with repetitions. In other words, if there are two edges between $P, Q \in G \wr \mathcal{P}$, then $\text{nb}(Q)$ will contain P twice.

Proposition 4. *Let $\mathcal{P} = \{P_1, \dots, P_p\} \cup \{Q_1, \dots, Q_q\}$ be an equitable partition of the variables and factors of the MRF specified by $I = (G, \theta)$. Let $G \wr \mathcal{P}$ be the factor quotient of I . Then, LMAP-LP(I) can be written as:*

$$\mu'^* = \operatorname{argmax}_{\mu' \in L'(G)} \sum_{P \in V(G \wr \mathcal{P})} \theta_P |P| \mu_P + \sum_{Q \in F(G \wr \mathcal{P})} \theta_Q |Q| \mu_Q.$$

Where θ_P, θ_Q are the parameters of I for the corresponding partition elements. The constraints $L'(G)$ are defined as the set of $\mu \geq 0$ such that:

$$\left\{ \begin{array}{l} \forall P, P' \in V(G \wr \mathcal{P}), Q \in F(G \wr \mathcal{P}) \\ \text{s.t. } P, P' \in \text{nb}(Q) : \\ \alpha'(Q) \equiv \mu_Q \leq \mu_P ; \beta'(Q) \equiv \mu_Q \leq \mu_{P'} ; \\ \gamma'(Q) \equiv \mu_P + \mu_{P'} - \mu_Q \leq 1 \end{array} \right\} \quad (3)$$

Proof. We omit a detailed proof of this proposition due to space restrictions. Essentially, the argument is that the reformulation of Sec. 7 in [3] holds for any equitable partition, not just the orbit partition of an MRF. In this case, the Q -classes generalize edge orbits while the P -classes generalize variable orbits. \square

This proposition tells us that we can construct $L'(G)$ by the following procedure: (1) we instantiate an LP variable μ_P for every variable class $P \in V(G \wr \mathcal{P})$ (i.e., every supervariable) (2) we instantiate an LP variable μ_Q for every factor class $Q \in F(G \wr \mathcal{P})$ (i.e. superfactor); (3) for every pair

Algorithm 1: Solving MRF I using an equivalent MRF J

Input: Ground MRF I and LP-equivalent MRF J **Output:** MAP-LP(I) solution μ

- 1 **Solve** MRF J , i.e., compute $\tau = \operatorname{argmax}_{\mu} \text{MAP-LP}(J)$;
 - 2 **Lift** the solution τ to LMAP-LP(J). That is, compute $\tau' = \hat{B}_P^J \tau$ (Thm. 3(b));
 - 3 **Recover** solution of I , i.e., compute $\mu = B_P^I \tau'$ (Thm. 3(a));
-

of classes P, Q , if some ground variable $x_i \in P$ is adjacent to some ground factor $\phi_{ij} \in Q$, we add the constraint $\mu_Q \leq \mu_P$. For every triplet P, P', Q such that there exist $x_i \in P, x_j \in P'$ adjacent to $\phi_{ij} \in Q$, we add the constraint $\mu_P + \mu_{P'} - \mu_Q \leq 1$.

Observe that the factor quotient graph $G \wr \mathcal{P}$ actually contains exactly the necessary and sufficient information to construct $L'(G)$: it gives us the number of classes and the relations between them. Hence, it would seem that $L'(G)$ is just $L(G \wr \mathcal{P})$, and we are done. Unfortunately this is not exactly the case, and we have to be a little bit more careful.

Recall our running example from Fig. 2. There can be a factor ϕ_{ij} in some Q , whose adjacent variables x_i, x_j fall into the same class, $P = P'$. In terms of constraints, the corresponding triplet P, P', Q , with $P = P'$ yields the constraint $2\mu_P - \mu_Q \leq 1$. Graphically, this situation occurs whenever $G \wr \mathcal{P}$ contains a double edge. This also happens in our running examples (see Fig. 2(b)). Unfortunately, such constraints have no analogue in MAP-LP(I)!

How can we deal with this? Assume for the moment that for any ground factor ϕ_{ij} , $\mathcal{P}(i) \neq \mathcal{P}(j)$, in other words $G \wr \mathcal{P}$ happens to be a simple graph (no edge connects at both ends to the same vertex, and there is no more than one edge between any two different vertices). Then we can compute a new weight vector $\theta' \in \mathbb{R}^q$ as $\theta'_Q = |Q|\theta_Q$, $\theta'_P = |P|\theta_P$ (cf. Eq. 3). In this case, the MRF $I' = (G \wr \mathcal{P}, \theta')$ would indeed be a smaller MRF, whose MAP-LP is identical to the LMAP-LP of I . This enables us to view lifting as reparametrization: (1) we compute $G \wr \mathcal{P}$ from G ; (2) instead of solving LMAP-LP(I), we solve MAP-LP(I') using any solver we want, including message-passing algorithms such as MPLP, TRWBP, among others; (3) because of the equivalence, we treat the solution of MAP-LP(I') as a solution of LMAP-LP(I) and unlift it using Thm. 3(a).

While our assumption does not hold in general (see e.g. Fig. 1) — and we will indeed account for it below — the procedure just outlined above is the main idea underlying “lifting by reparametrization” method. Since the LMAP-LP of I will potentially contain constraints such as $2\mu_P - \mu_Q \leq 1$, it will not be the MAP-LP of any simple graph. So instead, we will look for something

else, namely a proper (potentially much smaller) MRF J , where instead of LMAP-LP(I) = MAP-LP(J) we ask that LMAP-LP(I) = LMAP-LP(J). We call any pair of MRF where this holds LP-equivalent.

Definition 5 (LP equivalent MRFs). Two MRFs $I = (G, \theta^I)$ and $J = (H, \theta^J)$ having simple graphs are LP-equivalent if we can find an equitable partition \mathcal{P} of G with incidence matrix $B = (B_P, B_Q)$ and an equitable partition \mathcal{P}' of H with incidence matrix $B' = (B'_P, B'_Q)$ such that $\text{LMAP-LP}(I) := (\hat{B}_Q^T A B_P, \hat{B}_Q^T \mathbf{b}, B_P^T \mathbf{c}) = ((\hat{B}'_Q)^T A' B'_P, (\hat{B}'_Q)^T \mathbf{b}', (B'_P)^T \mathbf{c}') =: \text{LMAP-LP}(J)$.

Then, we apply the lifted equivalence of Thm. 3(b) and are done. As summarized in Alg. 1, we solve the smaller MAP-LP(J) using any MRF-structure-aware LP solver. We obtain an optimal solution of LMAP-LP(J) using B_P^T , as prescribed by Thm. 3(b). Due to the lifted equivalence, this solution is also a solution of LMAP-LP(I), hence we recover (or “unlift”) the solution with respect to I using B_P . In doing so, we end up with an optimal solution of MAP-LP(I). This procedure is outlined in Fig. 3. We will shortly prove its soundness.

Theorem 6. Let I and J be two LP-equivalent MRFs of possibly different sizes. Then, (A) if τ is feasible in MAP-LP(J), $\mu = B_P \hat{B}'_P \tau$ is feasible in MAP-LP(I). Moreover, if τ is optimal, μ is optimal as well. (B) if μ is feasible in MAP-LP(I), $\tau = B'_P \hat{B}_P \mu$ is feasible in MAP-LP(J). Moreover, if μ is optimal, τ is optimal as well.

Proof. We prove only (A) due to the symmetry of the statement. Let τ be feasible in MAP-LP(J). By Thm. 3(b), $\tau' = B'_P \tau$ is feasible in LMAP-LP(J). Due to LP-equivalence, LMAP-LP(J) = LMAP-LP(I), τ' is also a solution to LMAP-LP(I). Now, we unlift τ' with respect to LMAP-LP(I). Due to Thm. 3(b), $\mu = B_P (\hat{B}'_P)^T \tau'$ is feasible in MAP-LP(I). Moreover, if τ is optimal in MAP-LP(J), Thm. 3 tells us that optimality will hold throughout the entire chain of LPs. \square

To summarize our argument so far, Thm. 6 provides us with a way to exploit the MAP-LP equivalence between MRFs of different sizes. What is still missing is a way to efficiently construct such smaller LP-equivalent MRFs as input to Alg. 1. We will now address this issue.

Finding equivalent MRFs. So far we discussed the equivalence of MRFs of different sizes in terms of their (lifted) MAP-LPs. Making use of our result, however, requires efficient algorithm to find LP-equivalent MRFs of considerably smaller size. Given an MRF I and its EP, Alg. 2 finds the smallest LP-equivalent MRF I' in linear time. Next to illustrating Alg. 2 and proving that it is sound, we will also show that the size of I' is at most $2|G \wr \mathcal{P}|$.

Let $I = (G, \theta)$ be an MRF and \mathcal{P} be an EP of its variables and factors. We will introduce the algorithm in two

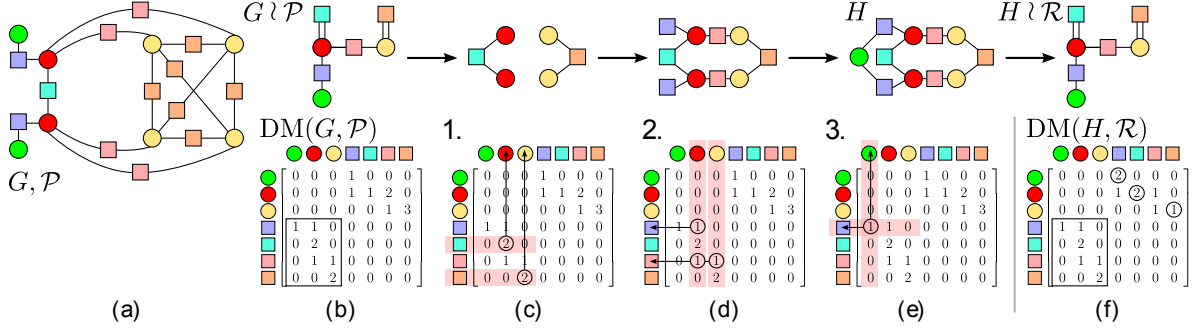


Figure 4: Illustration of “lifted inference by reparametrization” as summarized in Alg. 2 and used as input to Alg. 1. (a) input: a graph G and its coarsest equitable partition \mathcal{P} ; (b) the factor quotient $G \wr \mathcal{P}$ and the corresponding degree matrix; (c) **Step (A)**: representing all degree 2 factors and the corresponding degree 2 variables. (d) **Step (B)**: instantiating degree 1 factors adjacent to degree 2 variables; (e) **Step (C)**: instantiating the remaining classes. After this step we terminate and output the result H ; (f) soundness: $H \wr \mathcal{R}$ is identical to $G \wr \mathcal{P}$. Observe that the upper-right corner of $\text{DM}(G, \mathcal{P})$ is different from that of $\text{DM}(H, \mathcal{R})$. (Best viewed in color.)

steps: first, we discuss how to obtain a simple graph G' which gives the structure of an LP-equivalent MRF, i.e. $L'(G) = L'(G')$. Then we show how to find weights θ' for this graph, such that the MRF $I' = (G', \theta')$ is LP-equivalent to I . Finally, we will show correctness and minimality of our approach.

Recall that in Eq. 3 the lifted Ising polytope of LMAP-LP(I) is fully defined by the factor quotient $G \wr \mathcal{P}$. Hence, a necessary and sufficient condition for LP-equivalence (regarding the constraints of the LP, we will deal with the objective shortly) in MRFs is that the corresponding graphs exhibit the same factor quotients for some equitable partitions. Thus, the problem of finding an LP-equivalent structure boils down to finding G' such that $G \wr \mathcal{P} = G' \wr \mathcal{P}'$ for some partition \mathcal{P}' . Moreover, to maximize the compression, we want \mathcal{P} to be the coarsest EP of G (resulting in the smallest possible $G \wr \mathcal{P}$) and that G' is the smallest possible LP-equivalent MRF. Let us now see how to find G' . As a running example, we will use the factor graph in Fig 4(a).

Suppose $G \wr \mathcal{P}$ is given, e.g. computed using color-refinement. For our running example, $G \wr \mathcal{P}$ is shown in Fig. 4(b). Let us divide the superfactors and supervariables of $G \wr \mathcal{P}$ into classes based on their connectivity. A superfactor connected to a supervariable via a double edge is called a **(2)-superfactor**. In Fig. 4(b), these are the cyan and orange superfactors. Correspondingly, we call a variable connected to a superfactor via a double edge a **(2)-supervariable** (red and yellow in Fig. 4(b)). Next, a superfactor connected to at least one (2)-supervariable via a *single* edge is called a **(1, 2)-superfactor** (violet and pink in Fig. 4(b)). Finally, all other superfactors and supervariables are **(1)-superfactors** and **(1)-supervariables** respectively (e.g. the green supervariable).

We then compute G' in the following way as also illustrated in Fig. 4(c)-(e). We start with an empty graph. Then, **Step**

(A) as illustrated in Fig. 4(c) consists of adding for every (2)-superfactor in $G \wr \mathcal{P}$ exactly one representative factor to G' . Furthermore, for every (2)-supervariable, we add two representatives in G' and connect them to the corresponding (2)-superfactor representatives whenever the supernodes they represent are connected in $G \wr \mathcal{P}$. In **Step (B)**, see Fig. 4(d), for every (1, 2)-superfactor, we instantiate two representatives. Moreover, for every (2)-supervariable (all of them are already represented in G'), we match the two (1, 2)-superfactor representatives to the two (2)-supervariable representatives whenever the represented supernodes are connected in $G \wr \mathcal{P}$. Finally, **Step (C)** as shown in Fig. 4(e) introduces one representative for every other supernode and connects it to other representatives based on $G \wr \mathcal{P}$. If it happens that the represented supernode is connected to a (2)-supervariable or (1, 2)-superfactor in $G \wr \mathcal{P}$, we connect the representative to both representatives of the corresponding neighbor.

This is summarized in Alg. 2 and provably computes a minimal structure of an LP-equivalent MRF. Finally, we must compute a parameter vector for I' to facilitate LP-equivalence. Suppose \mathcal{P}' is the EP of G' induced by Alg. 2 (the partition which groups nodes in G' together if they represent the same supernode of $G \wr \mathcal{P}$). Let Q be any factor class in \mathcal{P} and Q' be the corresponding class in \mathcal{P}' . We then compute the weight $\theta_{Q'}$ of the factors $\phi' \in Q'$ of I' as

$$\theta_{Q'} = (|Q|/|Q'|)\theta_Q, \quad (4)$$

where θ_Q is the weight associated with the class Q in \mathcal{P} (recall Prop. 4). We now argue that the resulting Ising model $I' = (G', \theta')$ is LP-equivalent to $I = (G, \theta)$.

Theorem 7 (Soundness). $I' = (G', \theta')$ as computed above is LP-equivalent to $I = (G, \theta)$.

Proof. Following Def. 5 we must show that given I and its EP \mathcal{P} , there is a partition \mathcal{P}' of I' such that the lifted

Algorithm 2: Computing the smallest LP-equivalent MRF.**Input:** Fully lifted factor graph $G \wr \mathcal{P}$ of G **Output:** G' such that $\exists \mathcal{P}' : G' \wr \mathcal{P}' = G \wr \mathcal{P}$.

```

1 Initialize  $G' \leftarrow \emptyset$ , i.e., the empty graph;
  /* Step (A) Treat double edges */
2 for every (2)-superfactor  $Q$  in  $G \wr \mathcal{P}$  with neighboring
  (2)-supervariable  $P$  do
3   Add a factor  $q$  representing  $Q$  to  $G'$ ;
4   Add two variables  $p, p'$  representing  $P$  in  $G'$  and
   connect them to the factor  $q$ ;
5 end
  /* Step (B) To preserve degrees, treat now single
   edges in  $G'$  incident to double edges */
6 for every (1,2)-superfactor  $Q$  in  $G \wr \mathcal{P}$  do
7   Add two factors  $q, q'$  representing  $Q$  to  $G'$ ;
8   Connect  $q$  to  $p$  and  $q'$  to  $p'$  where  $p, p'$  are the
   representatives of a (2)-supervariable  $P$  in  $G \wr \mathcal{P}$ 
   that is neighboring  $Q$ ;
9 end
  /* Step (C) Add remaining nodes and edges to  $G'$  */
10 for all supervariables  $P$  and superfactor  $Q$  in
    $G \wr \mathcal{P}$  not represented in  $G'$  so far do
11   Add a single variable  $p$  resp. factor  $q$  to  $G'$ ;
12   Connect  $p$  to all representatives of superfactor  $Q$ 
   neighboring to  $P$  in  $G \wr \mathcal{P}$ ;
13 end

```

LPs are equal. We take the partition \mathcal{P}' to be the one induced by Alg. 2. \mathcal{P}' is equitable on G' by construction: we can go through Alg. 2 to verify that every two nodes in G' representing the same supernode of $G \wr \mathcal{P}$ are connected to the same number of representatives of every other supernode of $G \wr \mathcal{P}$ (we omit this due to space restrictions). Now, to show that LMAP-LP(I) has the same constraints as LMAP-LP(I'), we need $G \wr \mathcal{P} = G' \wr \mathcal{P}'$. To see that this holds, observe that Alg. 2 connects p to q in G' if only if P is connected to Q in $G \wr \mathcal{P}$: if Q is a (2)-superfactor, P is a (2)-supervariable – q will be connected to p in **Step (A)**. If P is a (2)-supervariable and Q is (1, 2)-superfactor, p and q will be connected in **Step (B)**. If Q is (1, 2)- of a (1)-superfactor and P is a (1)-supervariable, p and q will be connected in **Step (C)**. There are no other possible combinations. Hence, as \mathcal{P}' consists of all representatives of P and Q' consists of all representatives of Q , \mathcal{P}' and Q' are connected in $G' \wr \mathcal{P}'$ iff P is connected to Q . Moreover, representatives of (2)-superfactors are the only ones connected to two representatives of the same supervariable in G' , hence Q' is connected to P' via a double edge in $G' \wr \mathcal{P}'$ if and only if Q is connected to P via a double edge in $G \wr \mathcal{P}$.

Next, we argue that the objectives of the lifted LPs are the same. Using the parameters calculated with Eq. 4, the objective of LMAP-LP(I') is $\sum_{Q' \in \mathcal{P}'} |Q'| \theta_{Q'} \mu_{Q'} = \sum_{Q' \in \mathcal{P}'} |Q'| (|Q|/|Q'|) \theta_{Q'} \mu_{Q'} = \sum_{Q' \in \mathcal{P}'} |Q| \theta_{Q'} \mu_{Q'} =$

$\sum_{Q \in \mathcal{P}} |Q| \theta_Q \mu_Q$. Observe that the final term is exactly the objective of LMAP-LP(I) as given by Prop. 4. We conclude LMAP-LP(I) = LMAP-LP(I'). \square

We have thus shown that Alg. 2 and Eq. 4 together produce an LP-equivalent MRF. We will now show that this MRF is the smallest LP-equivalent MRF to the original.

Theorem 8 (Minimality). *Let $I = (G, \theta)$ be an Ising model and an $I' = (G', \theta')$ be computed as above. Then there is no other LP-Equivalent MRF with less factors or less vertices than G' . Moreover, $|V(G')| \leq 2|V(G \wr \mathcal{P})|$ and G' and $|E(G')| \leq 2|E(G \wr \mathcal{P})|$, i.e., the size of I' is at most twice the size of the fully lifted model.*

Proof. Let H be any graph with the same factor quotient as G . Then, let Q be a (2)-superfactor in $G \wr \mathcal{P}$ adjacent to some (2)-supervariable P . Due to equivalence, Q' is a (2)-superfactor in $H \wr \mathcal{P}'$ as well and P' is a (2)-supervariable. Hence, the class $P' \in \mathcal{P}'$ must have at least two ground variables from H . Next, let Q be a (1, 2)-factor in $G \wr \mathcal{P}$ adjacent to a (2)-supervariable. Analogously, Q' is a (1, 2)-factor in $H \wr \mathcal{P}'$ and P' is a (2)-supervariable. As we have established P' must have at least two ground elements in H . Since P' is connected to Q' via a single edge, the same holds on the ground level: any $p \in P'$ is connected to $q \in Q'$ via a single edge. This means that there are at least as many $q \in Q'$ as there are $p \in P'$, that is, at least two. All other supernodes must have at least one representative. These conditions are necessary for any LP-equivalent H .

Now, let G' be computed from Alg. 2 and \mathcal{P}' be the corresponding partition. To see why G' is minimal, observe that G' has *exactly* two representatives of any (2)-supervariable in $G \wr \mathcal{P}$ (step 1) and *exactly* two representatives of any (1, 2)-superfactor (step 2). All other supernodes have *exactly* one representative (steps 1 and 3). Therefore, G' meets the conditions with equality and is thus minimal. Finally, since we represent any supernode of $G \wr \mathcal{P}$ by at most 2 nodes in G' , G' can have at most twice as many factors and variables as $G \wr \mathcal{P}$. \square

Since Alg. 2 makes only one pass over the lifted factor graph, the overall time to compute the LP-equivalent MRF (which is then input to Alg. 1) is dominated by color-refinement, which is quasi-linear in the size of G .

4 EMPIRICAL ILLUSTRATION

The empirical illustration of our theoretical results investigates two questions. **(Q1)** Is reparametrization comparable to lifted MAP-LPs in terms of time and quality when using LP solvers? **(Q2)** If so, can lifted MPLP and TRW by reparametrization pay-off when solving MAP-LPs? And finally, **(Q3)** how does reparametrization behave in the presence of approximate evidence?

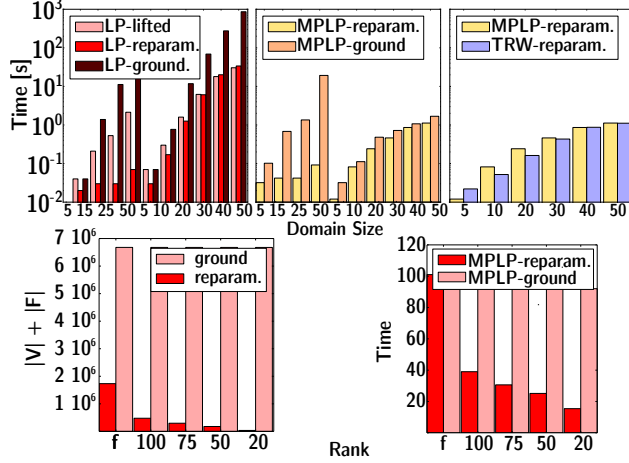


Figure 5: Experimental illustration (for ease of comparison the first three plots have the same scales). From top to bottom, from left to right: (a) Comparison (domain size vs. end-to-end running time) of solving LPs using GLPK (ground, fully lifted, and reparametrization). The first block of domain sizes (5, 15, 25, 50) are from the friends-smoker MLN; the second block (5, 10, . . . , 50) is on the CORA MLN. (b) Performance of MPLP (ground vs. reparametrization) on the same MLNs (same block structure). (c) Comparison (domain size vs. end-to-end running time) of TRW and MPLP by reparametrization on CORA. (d,e) Model sizes for exact evidence (“f”) and approximations of ranks 100 to 20 and running times.

To this aim we implemented the reparametrization approach on a single Linux machine (4 × 3.4 GHz cores, 32 GB main memory) using Python and C/C++. For evaluation we considered three sets of MRFs. One was generated from grounding a modified version of a Markov Logic Network (MLN) used for entity resolution on the CORA dataset. Five different MRFs were generated by grounding the model for 5, 10, 20, 30, 40 and 50 entities, having 960, 4081, 13933, 27850, 4699 and 76274 factors respectively. The second set was generated from a pairwise version of the friends-smokers MLN [4] for 5, 15, 25 and 50 people, having 190, 1620, 4450 and 17650 factors. The third set considers a simple $\text{fr}(X, Y) \Rightarrow (\text{sm}(X) \Leftrightarrow \text{sm}(Y))$ rule (converted to a pairwise MLN) where we used the `link_common` observations from the “Cornell” dataset as evidence for `fr`. Then we computed different low-rank approximations of the evidence using [23].

In all cases, there were only few additional factors due to treating double edges. What is more interesting are the running times and overall performances. Fig. 5(a) shows the end-to-end running time for solving the corresponding ground, (fully) lifted, and reparametrized LPs using GLPK. As one can see, reparametrization is competitive to lifted linear programming (LLP) in time. Actually, it can even save time since it runs directly on the factor graph and

not on the LP matrix — which is larger than the factor graph — for discovering symmetries. Moreover, in all cases the same objective was achieved, that is, reparametrization does not sacrifice quality. In turn, question (Q1) can clearly be answered affirmatively. Fig. 5(b) summarizes the performance of MPLP on the reparametrized models. As one can see, MPLP can be significantly faster than LLP for solving MAP-LPs without sacrificing the objective; it was always identical to the LP solutions. To illustrate than one may also run other LP-based message-passing solvers, Figs. 5(c) summarizes the performance of TRW on CORA. As one can see, lifting TRW by reparametrization is possible and differences in time are likely due to initialization, stopping criterion, etc. In any case, question (Q2) can clearly be answered affirmatively. All results so far show that lifted LP-based MP solvers can be significantly faster than generic LP solvers. Figs. 5(d,e) summarize the results for low-rank evidence approximation. As one can see in (d), significant reduction in model size can be achieved even at rank 100, which in turn can lead to faster MPLP running times (e). For each low-rank model, the ground and the reparametrized MPLP achieved the same objective. Plot (e), however, omits the time for performing BMF. It can be too costly to first run BMF canceling the benefits of lifted LP-based inference (in contrast to exact inference as in [23]). Nevertheless, w.r.t. (Q3) these results illustrate that evidence approximation can result in major speed-ups.

5 CONCLUSIONS

In this paper, we proved that lifted MAP-LP inference in MRFs with symmetries can be reduced to MAP-LP inference in standard models of reduced size. In turn, we can use any off-the-shelf MAP-LP inference algorithm — in particular approaches based on message-passing — for lifted inference. This incurs no major overhead: for given evidence, the reduced MRF is at most twice as large than the corresponding fully lifted MRF. By plugging in different existing MAP-LP inference algorithms, our approach yields a family of lifted MAP-LP inference algorithms. We illustrated this empirically for MPLP and tree-reweighted BP. In fact, running MPLP yields the first provably convergent lifted MP approach for MAP-LP relaxations. More importantly, our result suggests a novel view on lifted inference: *lifted inference can be viewed as standard inference in a reparametrized model*. Exploring this view for marginal inference as well as for branch-and-bound MAP inference approaches are the most attractive avenue for future work.

Acknowledgments: The authors would like to thank the anonymous reviewers and Udi Apsel for their feedback and Guy Van den Broeck for providing the evidence approximation code. This research was partly supported by the German-Israeli Foundation (GIF) for Scientific Research and Development, 1180-218.6/2011, and by the German Science Foundation (DFG), KE 1686/2-1.

References

- [1] B. Ahmadi, K. Kersting, M. Mladenov, and S. Natarajan. Exploiting symmetries for scaling loopy belief propagation and relational training. *Machine Learning*, 92(1):91–132, 2013.
- [2] C. Berkholz, P. Bonsma, and M. Grohe. Tight lower and upper bounds for the complexity of canonical colour refinement. In *Proceedings of the 21st Annual European Symposium on Algorithms*, 2013.
- [3] H.H. Bui, T.N. Huynh, and S. Riedel. Automorphism groups of graphical models and lifted variational inference. In *Proc. of the 29th Conference on Uncertainty in Artificial Intelligence (UAI-2013)*, 2013.
- [4] D. Fierens, K. Kersting, J. Davis, J. Chen, and M. Mladenov. Pairwise markov logic. In *Postproc. of the 22nd International Conference on Inductive Logic Programming (ILP-2012)*, 2012.
- [5] L. Getoor and B. Taskar, editors. *An Introduction to Statistical Relational Learning*. MIT Press, 2007.
- [6] A. Globerson and T. Jaakkola. Fixing max-product: Convergent message passing algorithms for map LP-relaxations. In *Proc. of the 21st Annual Conf. on Neural Inf. Processing Systems (NIPS)*, 2007.
- [7] M. Grohe, K. Kersting, M. Mladenov, and E. Selman. Dimension reduction via colour refinement. *CoRR*, abs/1307.5697, 2013.
- [8] F. Hadiji and K. Kersting. Reduce and re-lift: Bootstrapped lifted likelihood maximization for map. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence (AAAI)*, 2013.
- [9] K. Kersting. Lifted probabilistic inference. In *Proceedings of ECAI-2012*. IOS Press, 2012.
- [10] K. Kersting, B. Ahmadi, and S. Natarajan. Counting Belief Propagation. In *Proc. of the 25th Conf. on Uncertainty in Artificial Intelligence (UAI-09)*, 2009.
- [11] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.
- [12] O. Meshi, T. Jaakkola, and A. Globerson. Convergence rate analysis of MAP coordinate minimization algorithms. In *Advances in Neural information Processing Systems 25*. MIT Press, 2012.
- [13] M. Mladenov, B. Ahmadi, and K. Kersting. Lifted linear programming. In *15th Int. Conf. on Artificial Intelligence and Statistics (AISTATS 2012)*, pages 788–797, 2012. Volume 22 of JMLR: W&CP 22.
- [14] M. Mladenov, A. Globerson, and K. Kersting. Efficient lifting of map lp relaxations using k -locality. In *17th Int. Conf. on Artificial Intelligence and Statistics (AISTATS 2014)*, 2014. JMLR: W&CP volume 33.
- [15] J. Noessner, M. Niepert, and H. Stuckenschmidt. Rockit: Exploiting parallelism and symmetry for map inference in statistical relational models. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence (AAAI)*, 2013.
- [16] D. Poole. First-Order Probabilistic Inference. In *Proc. of the 18th International Joint Conference on Artificial Intelligence (IJCAI-05)*, pages 985–991, 2003.
- [17] M. Richardson and P. Domingos. Markov Logic Networks. *Machine Learning*, 62:107–136, 2006.
- [18] S. Sarkhel, D. Venugopal, P. Single, and V. Gogate. Lifted map inference for markov logic networks. In *17th Int. Conf. on Artificial Intelligence and Statistics (AISTATS 2014)*, 2014. JMLR: W&CP volume 33.
- [19] P. Singla and P. Domingos. Lifted First-Order Belief Propagation. In *Proc. of the 23rd AAAI Conf. on Artificial Intelligence (AAAI-08)*, pages 1094–1099, Chicago, IL, USA, July 13-17 2008.
- [20] D. Sontag, D.K. Choe, and Y. Li. Efficiently searching for frustrated cycles in map inference. In *Proc. of the 28th Conference on Uncertainty in Artificial Intelligence (UAI-2012)*, pages 795–804, 2012.
- [21] D. Sontag and T. Jaakkola. New outer bounds on the marginal polytope. In *Proc. of the 21st Annual Conference on Neural Information Processing Systems (NIPS-2007)*, 2007.
- [22] G. Van den Broeck, A. Choi, and A. Darwiche. Lifted relax, compensate and then recover: From approximate to exact lifted probabilistic inference. In *Proc. of the 28th Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2012.
- [23] G. Van den Broeck and A. Darwiche. On the complexity and approximation of binary evidence in lifted inference. In *Proc. of the 27th Annual Conf. on Neural Information Processing Systems (NIPS)*, 2013.
- [24] M.J. Wainwright, T. Jaakkola, and A.S. Willsky. Map estimation via agreement on trees: message-passing and linear programming. *IEEE Transactions on Information Theory*, 51(11):3697–3717, 2005.
- [25] M.J. Wainwright and M.I. Jordan. Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.*, 1(1-2):1–305, January 2008.

Fast Gaussian Process Posteriors with Product Trees

David A. Moore
Computer Science Division
University of California, Berkeley
Berkeley, CA 94709
dmoore@cs.berkeley.edu

Stuart Russell
Computer Science Division
University of California, Berkeley
Berkeley, CA 94709
russell@cs.berkeley.edu

Abstract

Gaussian processes (GP) are a powerful tool for nonparametric regression; unfortunately, calculating the posterior variance in a standard GP model requires time $O(n^2)$ in the size of the training set. Previous work by Shen et al. (2006) used a k -d tree structure to approximate the posterior mean in certain GP models. We extend this approach to achieve efficient approximation of the posterior covariance using a tree clustering on pairs of training points, and demonstrate significant improvements in performance with negligible loss of accuracy.

1 INTRODUCTION

Complex Bayesian models often tie together many smaller components, each of which must provide its output in terms of probabilities rather than discrete predictions. Gaussian process (GP) regression (Rasmussen and Williams, 2006) is a natural fit for such systems, but its applications have been limited by computational concerns: training a GP model on n points requires $O(n^3)$ time, while computing the posterior distribution at a test point requires $O(n)$ and $O(n^2)$ operations for the mean and variance respectively.

This paper focuses on the fast evaluation of GP posterior probabilities in a running inference system, for which a model has already been trained. Fast runtime performance is a common requirement for real-world systems; for example, a speech recognition system might be trained once in the cloud, then run many times on a smartphone under a tight computational budget. Our particular work is motivated by an application to nuclear test monitoring: after training on historical seismic events, we want to identify and localize new events in realtime by processing signals from a worldwide sensor network. In this application, as with many others, probabilities from a GP are computed in the inner loop of a message-passing or MCMC

inference algorithm; this computation must be efficient if inference is to be feasible.

Previous work has explored the use of space-partitioning tree structures for efficient computation of GP posterior means in models where the covariance kernel has a short lengthscale or compact support (Shen et al., 2006). We extend this in several ways. First, we describe the *product tree* data structure, along with an algorithm that uses this structure to efficiently compute posterior covariances. This provides what is to our knowledge the first account of GP regression in which the major test-time operations (posterior mean and covariance) run in time sublinear in the training set size, given a suitably sparse kernel matrix. We give a novel cutoff rule, applicable to both mean and covariance calculations, that guarantees provably bounded error. We also extend the class of models to which tree-based methods can be efficiently applied, by showing how to include a low-rank global component modeled either by an explicit parametric representation or by an approximate GP with inducing points (Snelson and Ghahramani, 2006). Finally, we evaluate this work empirically, with results demonstrating significant speedups on synthetic and real-world data, and in the process identify a simple method that often provides competitive performance to the more complex product tree.

2 BACKGROUND

2.1 GP REGRESSION MODEL

We assume as training input a set of labeled points $\{(\mathbf{x}_i, y_i) | i = 1, \dots, n\}$, where we suppose that

$$y_i = f(\mathbf{x}_i) + \epsilon_i \quad (1)$$

for some unknown function $f(\cdot)$ and i.i.d. Gaussian observation noise $\epsilon_i \sim \mathcal{N}(0, \sigma_n^2)$. Treating the estimation of $f(\cdot)$ as a Bayesian inference problem, we consider a Gaussian process prior distribution $f(\cdot) \sim GP(0, k)$, parameterized by a positive-definite *covariance* or *kernel* function $k(x, x')$. Given a set X^* containing m test points, we de-

rive a Gaussian posterior distribution $f(X^*) \sim \mathcal{N}(\mu^*, \Sigma^*)$, where

$$\mu^* = K^{*T} K_y^{-1} \mathbf{y} \quad (2)$$

$$\Sigma^* = K^{**} - K^{*T} K_y^{-1} K^* \quad (3)$$

and $K_y = k(X, X) + \sigma_n^2 I$ is the covariance matrix of training set observations, $K^* = k(X, X^*)$ denotes the $n \times m$ matrix containing the kernel evaluated at each pair of training and test points, and similarly $K^{**} = k(X^*, X^*)$ gives the kernel evaluations at each pair of test points. Details of the derivations, along with general background on GP regression, can be found in Rasmussen and Williams (2006).

In this work, we make the additional assumption that the input points \mathbf{x}_i and test points \mathbf{x}_p^* lie in some metric space (\mathcal{M}, d) , and that the kernel is a monotonically decreasing function of the distance metric. Many common kernels fit into this framework, including squared-exponential, rational quadratic, piecewise-polynomial and Matérn kernel families; anisotropic kernels can be represented through choice of an appropriate metric.

2.2 RELATED WORK

Tree structures such as k -d trees (Friedman et al., 1977) form a hierarchical, multiresolution partitioning of a dataset, and are commonly used in machine learning for efficient nearest-neighbor queries. They have also been adapted to speed up nonparametric regression (Moore et al., 1997; Shen et al., 2006); the general approach is to view the regression computation of interest as a sum over some quantity associated with each training point, weighted by the kernel evaluation against a test point. If there are sets of training points having similar weight – for example, if the kernel is very wide, if the points are very close to each other, or if the points are all far enough from the query to have effectively zero weight – then the weighted sum over the set of points can be approximated by an unweighted sum (which does not depend on the query and may be precomputed) times an estimate of the typical weight for the group, saving the effort of examining each point individually. This is implemented as a recursion over a tree structure augmented at each node with the unweighted sum over all descendants, so that recursion can be cut off with an approximation whenever the weight function is shown to be suitably uniform over the current region.

This tree recursion can be thought of as an approximate matrix-vector multiplication (MVM) operation; a related method, the Improved Fast Gauss Transform (Morariu et al., 2008), implements fast MVM for the special case of the SE kernel. It is possible to accelerate GP training by combining MVM methods with a conjugate gradient solver, but models thus trained do not allow for the computation of predictive variances. One argument against MVM techniques (and, by extension, the approach of this paper)

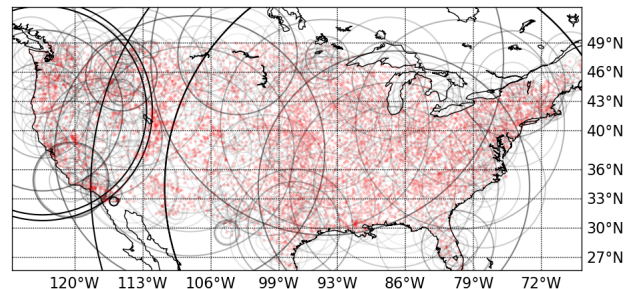


Figure 1: Cover tree decomposition of USA precipitation measurement stations (see Section 5.3).

is that their efficiency requires shorter lengthscales than are common in machine learning applications (Murray, 2009); however, we have found them quite effective on datasets which do have genuinely sparse covariance structure (e.g., geospatial data), or in which the longer-scale variation can be represented by a low-rank component.

Another related approach is the use of local approximations, in which different GPs are trained in different regions of the input space. There is some evidence that these can provide accurate predictions which are very fast to evaluate (Chalupka et al., 2013); however, they face boundary discontinuities and inaccurate uncertainty estimates if the data do not naturally form independent clusters.

2.3 k -d VERSUS COVER TREES

Although related work (Moore et al., 1997; Shen et al., 2006) has generally used k -d trees as the multiresolution structure, this paper instead uses cover trees (Beygelzimer et al., 2006) to allow for non-Euclidean metrics. A cover tree on n points can be constructed in $O(n \log n)$ time, and the construction and query times scale only with the *intrinsic* dimensionality of the data, allowing for efficient nearest-neighbor queries in higher-dimensional spaces (Beygelzimer et al., 2006). Figure 1 shows a cover-tree decomposition of one of our test datasets.

We do not depend specifically on the cover tree algorithm; any similar tree construction algorithm could be used, provided (a) there is a one-to-one correspondence between the leaves of the tree and the training points $x_i \in X$, and (b) each non-leaf node \mathbf{n} is associated with some point $x_{\mathbf{n}} \in \mathcal{M}$, such that all descendants of \mathbf{n} are contained within a ball of radius $r_{\mathbf{n}}$ centered at $x_{\mathbf{n}}$. For example, a ball tree (Uhlmann, 1991), or a tree created through agglomerative clustering on the training points, could also satisfy these criteria.

```

initialize globals sum  $S \leftarrow 0$ , error  $\epsilon \leftarrow 0$ , leaf count  $\kappa \leftarrow 0$ 
function WEIGHTEDMETRICSUM(node  $\mathbf{n}$ , query points  $(\mathbf{x}_i^*, \mathbf{x}_j^*)$ ,
                                tolerance  $\epsilon_{\text{abs}}$ )
     $\delta_{\mathbf{n}} \leftarrow \delta((\mathbf{x}_i^*, \mathbf{x}_j^*), (\mathbf{n}_1, \mathbf{n}_2))$ 
    if  $\mathbf{n}$  is a leaf then
         $S \leftarrow S + (K_y^{-1})_{\mathbf{n}} \cdot (k(d(\mathbf{x}_i^*, \mathbf{n}_1)) \cdot k(d(\mathbf{x}_j^*, \mathbf{n}_2)))$ 
         $\kappa \leftarrow \kappa + 1$ 
    else
         $w_{\min} \leftarrow k_{\text{lower}}^{\text{prod}}(\delta_{\mathbf{n}} + r_{\mathbf{n}})$ 
         $w_{\max} \leftarrow k_{\text{upper}}^{\text{prod}}(\max(\delta_{\mathbf{n}} - r_{\mathbf{n}}, 0))$ 
         $\epsilon_{\mathbf{n}} \leftarrow \frac{1}{2}(w_{\max} - w_{\min})S_{\mathbf{n}}^{\text{Abs}}$ 
        if  $\epsilon_{\mathbf{n}} \leq \kappa_{\mathbf{n}} / (n - \kappa) \cdot (\epsilon_{\text{abs}} - \epsilon)$  then
             $S \leftarrow S + \frac{1}{2}(w_{\max} + w_{\min}) \cdot S_{\mathbf{n}}^{\text{UW}}$ 
             $\kappa \leftarrow \kappa + \kappa_{\mathbf{n}}$ 
             $\epsilon \leftarrow \epsilon + \epsilon_{\mathbf{n}}$ 
        else
            for each child  $\mathbf{c}$  of  $\mathbf{n}$ 
                sorted by descending  $\delta((\mathbf{x}_i^*, \mathbf{x}_j^*), (\mathbf{c}_1, \mathbf{c}_2))$  do
                    WEIGHTEDMETRICSUM( $\mathbf{c}$ ,  $(\mathbf{x}_i^*, \mathbf{x}_j^*)$ ,  $\epsilon_{\text{abs}}$ )
            end for
        end if
    end if
end function

```

Figure 2: Recursive algorithm to computing GP covariance entries using a product tree. Abusing notation, we use \mathbf{n} to represent both a tree node and the pair of points $\mathbf{n} = (\mathbf{n}_1, \mathbf{n}_2)$ associated with that node.

3 EFFICIENT COVARIANCE USING PRODUCT TREES

We now consider efficient calculation of the GP covariance (3). The primary challenge is the multiplication $K^{*T}K_y^{-1}K^*$. For simplicity of exposition, we will focus on computing the (i, j) th entry of the resulting matrix, i.e., on the multiplication $\mathbf{k}_i^{*T}K_y^{-1}\mathbf{k}_j^*$ where \mathbf{k}_i^* denotes the vector of kernel evaluations between the training set and the i th test point, or equivalently the i th column of K^* . Note that a naïve implementation of this multiplication requires $O(n^2)$ time.

We might be tempted to apply the vector multiplication primitive of Shen et al. (2006) separately for each row of K_y^{-1} to compute $K_y^{-1}\mathbf{k}_j^*$, and then once more to multiply the resulting vector by \mathbf{k}_i^* . Unfortunately, this requires n vector multiplications and thus scales (at least) linearly in the size of the training set. Instead, we note that we can rewrite $\mathbf{k}_i^{*T}K_y^{-1}\mathbf{k}_j^*$ as a weighted sum of the entries of K_y^{-1} , where the weight of the (p, q) th entry is given by $k(\mathbf{x}_i^*, \mathbf{x}_p)k(\mathbf{x}_j^*, \mathbf{x}_q)$:

$$\mathbf{k}_i^{*T}K_y^{-1}\mathbf{k}_j^* = \sum_{p=1}^n \sum_{q=1}^n (K_y^{-1})_{pq} k(\mathbf{x}_i^*, \mathbf{x}_p) k(\mathbf{x}_j^*, \mathbf{x}_q). \quad (4)$$

Our goal is to compute this weighted sum efficiently using a tree structure, similar to Shen et al. (2006), except that instead of clustering points with similar weights, we now want to cluster *pairs* of points having similar weights.

To do this, we consider the *product space* $\mathcal{M} \times \mathcal{M}$ consisting of all pairs of points from \mathcal{M} , and define a *product*

metric δ on this space. The details of the product metric will depend on the choice of kernel function (section 3.2). For the moment, we will assume an SE kernel, of the form $k_{SE}(d) = \exp(-d^2)$, for which a natural choice is the 2-product metric:

$$\delta((\mathbf{x}_a, \mathbf{x}_b), (\mathbf{x}_c, \mathbf{x}_d)) = \sqrt{d(\mathbf{x}_a, \mathbf{x}_c)^2 + d(\mathbf{x}_b, \mathbf{x}_d)^2},$$

which has the fortunate property

$$k_{SE}(d(\mathbf{x}_a, \mathbf{x}_b))k_{SE}(d(\mathbf{x}_c, \mathbf{x}_d)) = k_{SE}(\delta((\mathbf{x}_a, \mathbf{x}_b), (\mathbf{x}_c, \mathbf{x}_d))),$$

i.e., the property that evaluating the SE kernel in the product space (the right hand side) gives us the correct weight for our weighted sum (4) (the left hand side). Note that this property is convenient but not necessary; Section 3.2 describes how to choose a product metric for several common kernels.

Now we can run any metric tree construction algorithm (e.g., a cover tree) using the product metric to build a *product tree* on all *pairs* of training points. At each leaf node \mathbf{L} , representing a pair of training points, we store the entry $(K_y^{-1})_{\mathbf{L}}$ corresponding to those two training points, and at each higher-level node \mathbf{n} we cache the unweighted sum $S_{\mathbf{n}}^{\text{UW}}$ of these entries over all of its descendant leaf nodes, as well as the sum of absolute values $S_{\mathbf{n}}^{\text{Abs}}$ (these cached sums will be used to determine when to cut off recursive calculations):

$$S_{\mathbf{n}}^{\text{UW}} = \sum_{\mathbf{L} \in \text{leaves}(\mathbf{n})} (K_y^{-1})_{\mathbf{L}} \quad (5)$$

$$S_{\mathbf{n}}^{\text{Abs}} = \sum_{\mathbf{L} \in \text{leaves}(\mathbf{n})} |(K_y^{-1})_{\mathbf{L}}|. \quad (6)$$

Given a product tree augmented in this way, the weighted-sum calculation (4) is approximated by the WEIGHTEDMETRICSUM algorithm of Figure 2. It proceeds by a recursive descent down the tree, where at each non-leaf node \mathbf{n} it computes upper and lower bounds on the weight of any descendant, and applies a cutoff rule (Section 3.1) to determine whether to continue the descent. Whenever the descent is halted, we approximate the contribution from leaves below \mathbf{n} by $\frac{1}{2}(w_{\max} + w_{\min}) \cdot S_{\mathbf{n}}^{\text{UW}}$, i.e., by the average weight times the unweighted sum. Otherwise, the computation continues recursively over \mathbf{n} 's children.

3.1 CUTOFF RULE

The decision of when to cut off the tree recursion is crucial to correctness and performance. Many cutoff rules are possible. For predictive mean calculation, Moore et al. (1997) and Shen et al. (2006) maintain an accumulated lower bound on the total overall weight, and cut off whenever the difference between the upper and lower weight bounds at the current node is a small fraction of the lower bound on the overall weight. By contrast, we introduce a

rule (8) which takes into account the weights as well as the entries of K_y^{-1} being summed over (since we expect this matrix to be approximately sparse, some entries will contribute much more to the sum than others), and which provides a provable guarantee on approximation error not available in the earlier work. First, at each node \mathbf{n} we compute an error bound

$$\epsilon_{\mathbf{n}} = \frac{1}{2}(w_{\max} - w_{\min})S_{\mathbf{n}}^{\text{Abs}}, \quad (7)$$

which we justify by the following lemma:

Lemma 1. *The error introduced in approximating the subtree at \mathbf{n} by $\frac{1}{2}(w_{\max} + w_{\min})S_{\mathbf{n}}^{\text{UW}}$ is bounded by $\epsilon_{\mathbf{n}}$.*

Proof. Let $S_{\mathbf{n}}^+$ and $S_{\mathbf{n}}^-$ denote the sum of positive and negative leaf entries under \mathbf{n} , respectively, so $S_{\mathbf{n}}^{\text{UW}} = (S_{\mathbf{n}}^+ + S_{\mathbf{n}}^-)$ and $S_{\mathbf{n}}^{\text{Abs}} = (S_{\mathbf{n}}^+ - S_{\mathbf{n}}^-)$. The worst case for the approximation is if the true sum gives weight w_{\max} to $S_{\mathbf{n}}^+$ and w_{\min} to $S_{\mathbf{n}}^-$ (or vice versa), yielding an approximation error of

$$\left| (w_{\max}S_{\mathbf{n}}^+ + w_{\min}S_{\mathbf{n}}^-) - \frac{w_{\max} + w_{\min}}{2}S_{\mathbf{n}}^{\text{UW}} \right| = \epsilon_{\mathbf{n}}.$$

□

Intuitively, we see that $\epsilon_{\mathbf{n}}$ is small whenever the leaves below \mathbf{n} have nearly uniform weights, or when the total mass of K_y^{-1} entries under \mathbf{n} is small. This motivates our cutoff rule

$$\epsilon_{\mathbf{n}} \leq \kappa_{\mathbf{n}} / (n - \kappa) \cdot (\epsilon_{\text{abs}} - \epsilon), \quad (8)$$

in which ϵ_{abs} is a user-specified bound on the absolute error of the overall computation, $\kappa_{\mathbf{n}}$ denotes the number of leaves below \mathbf{n} , n and κ denote respectively the total number of leaves in the tree and the leaves included thus far in the partially computed sum, and $\epsilon = \sum_{\mathbf{n} \in C} \epsilon_{\mathbf{n}}$, where C is the set of all intermediate nodes whose leaf sums we have previously approximated, is a running upper bound on the total error accumulated thus far in the sum. Intuitively, $(\epsilon_{\text{abs}} - \epsilon)$ gives the “error budget” remaining out of an initial budget of ϵ_{abs} ; each cutoff is allowed to use a fraction of this budget proportional to the number of leaves being approximated. We show the following correctness result:

Theorem 1. *Let \mathbf{T} be a product tree constructed on K_y^{-1} , where $\hat{\Sigma}_{ij}^* = K_{ij}^{**} - \text{WEIGHTEDMETRICSUM}(\mathbf{T}, (\mathbf{x}_i^*, \mathbf{x}_j^*), \epsilon_{\text{abs}})$ denotes the approximation returned by the tree recursion to the true posterior covariance Σ_{ij}^* . Then $|\Sigma_{i,j}^* - \hat{\Sigma}_{i,j}^*| \leq \epsilon_{\text{abs}}$.*

Proof. By our cutoff rule (8), we proceed with the approximation at \mathbf{n} only if $\epsilon' := \epsilon_{\mathbf{n}} + \epsilon \leq \epsilon_{\text{abs}}$, i.e. only if the new error being introduced (bounded by Lemma 1), plus the error already accumulated, is still bounded by ϵ_{abs} .¹ Thus at

¹We have ignored the factor $\kappa_{\mathbf{n}} / (n - \kappa)$ here since it is always ≤ 1 ; this factor is included to help “pace” the computation and is not necessary for correctness.

every step we maintain the invariant $\epsilon \leq \epsilon_{\text{abs}}$, which establishes the result. □

Remark. *Although Theorem 1 bounds absolute error, we can also apply it to bound relative error in the case of computing a predictive variance that includes a noise component. Since the noise variance σ_n^2 is a lower bound on the predictive variance Σ_{ii}^* , setting $\epsilon_{\text{abs}} = \epsilon_{\text{rel}} \cdot \sigma_n^2$ is sufficient to ensure that the approximation error is smaller than $\epsilon_{\text{rel}} \cdot \Sigma_{ii}^*$.*

Note that this cutoff rule and correctness proof can be easily back-ported into the WEIGHTEDSUM algorithm of Shen et al. (2006), providing a bounded error guarantee for tree-based calculations of GP posterior means as well as covariances.

3.2 OTHER KERNEL FUNCTIONS

As noted above, the SE kernel has the lucky property that, if we choose product metric $\delta = \sqrt{d_1^2 + d_2^2}$, then the product of two SE kernels is equal to the kernel of the product metric δ :

$$k_{SE}(d_1)k_{SE}(d_2) = \exp(-d_1^2 - d_2^2) = k_{SE}(\delta).$$

In general, however, we are not so lucky: it is not the case that every kernel we might wish to use has a corresponding product metric such that a product of kernels can be expressed in terms of the product metric. In such cases, we may resort to upper and lower bounds in place of computing the exact kernel value. Note that such bounds are all we require to evaluate the error bound (7), and that when we reach a leaf node representing a specific pair of points we can always evaluate the exact product of kernels directly at that node.

For example, consider the kernel $k_{CS,0}(d) = (1 - d)_+^j$ (taking $j = \lfloor \frac{D}{2} \rfloor + 1$, where D is the input dimension); this is a simple example of a more general class of piecewise-polynomial compactly supported kernels (Rasmussen and Williams, 2006) whose computational advantages are especially relevant to tree-based algorithms. Considering the product of two such kernels,

$$k_{CS,0}(d_1)k_{CS,0}(d_2) = (1 - (d_1 + d_2) + \Delta)_+^j$$

where $\Delta = d_1d_2$ if $(d_1 < 1, d_2 < 1)$ else 0

we notice that this is almost equivalent to $k_{CS,0}(\delta)$ for the choice of $\delta = d_1 + d_2$, but with an additional pairwise term Δ . We bound this term by noting that it is maximized when $d_1 = d_2 = \delta/2$ (for $\delta < 2$) and minimized whenever either $d_1 = 0$ or $d_2 = 0$, so we have $(\delta/2)^2 \geq \Delta \geq 0$. This yields the bounds $k_{\text{lower}}^{\text{prod}}$ and $k_{\text{upper}}^{\text{prod}}$ as shown in Table 1. Bounds for other common kernels are obtained analogously in Table 1.

3.3 OPTIMIZATIONS

A naïve product tree on n points will have n^2 leaves, but we can reduce this and achieve substantial speedups by

Kernel	$k(d)$	$k(d_1)k(d_2)$	$\delta(d_1, d_2)$	$k_{\text{lower}}^{\text{prod}}(\delta)$	$k_{\text{upper}}^{\text{prod}}(\delta)$
SE	$\exp(-d^2)$	$\exp(-d_1^2 - d_2^2)$	$\sqrt{d_1^2 + d_2^2}$	$\exp(-(\delta)^2)$	$\exp(-(\delta)^2)$
γ -exponential	$\exp(-d^\gamma)$	$\exp(-d_1^\gamma - d_2^\gamma)$	$(d_1^\gamma + d_2^\gamma)^{1/\gamma}$	$\exp(-(\delta)^\gamma)$	$\exp(-(\delta)^\gamma)$
Piecewise polynomial CS _{D, q=0} , $j = \lfloor \frac{D}{2} \rfloor + 1$	$(1-d)_+^j$	$(1 - (d_1 + d_2) + \Delta)_+^j$ where $\Delta = d_1 d_2$ if $(d_1 < 1, d_2 < 1)$ else 0	$d_1 + d_2$	$(1-\delta)_+^j$	$\left(1 - \delta + \frac{(\delta)^2}{4}\right)_+^j$
Rational Quadratic	$\left(1 + \frac{d^2}{2\alpha}\right)^{-\alpha}$	$\left(1 + \frac{d_1^2 + d_2^2}{2\alpha} + \frac{d_1^2 d_2^2}{4\alpha^2}\right)^{-\alpha}$	$\sqrt{d_1^2 + d_2^2}$	$\left(1 + \frac{(\delta)^2}{2\alpha} + \frac{(\delta)^4}{16\alpha^2}\right)^{-\alpha}$	$\left(1 + \frac{(\delta)^2}{2\alpha}\right)^{-\alpha}$
Matérn ($\nu = 3/2$)	$\frac{1 + \sqrt{3}d}{2} \exp(-\sqrt{3}d)$	$\frac{1 + \sqrt{3}(d_1 + d_2) + 3d_1 d_2}{2} \exp(-\sqrt{3}(d_1 + d_2))$	$d_1 + d_2$	$\frac{1 + \sqrt{3}\delta}{2} \exp(-\sqrt{3}\delta)$	$\frac{1 + \sqrt{3}\delta + 3(\delta/2)^2}{2} \exp(-\sqrt{3}\delta)$

Table 1: Bounds for products of common kernel functions, all from Rasmussen and Williams (2006).

exploiting the structure of K_y^{-1} and of the product space $\mathcal{M} \times \mathcal{M}$:

Sparsity. If K_y is sparse, as with compactly supported kernel functions, or can be well-approximated as sparse, as when using standard kernels with short lengthscales, then it can be shown that K_y^{-1} may also be approximated as sparse (Bickel and Lindner, 2012, sections 2 and 4.1). When this is the case, the tree need include only those pairs $(\mathbf{x}_p, \mathbf{x}_q)$ for which $(K_y^{-1})_{pq}$ is non-negligible.

Symmetry. Since K_y^{-1} is a symmetric matrix, it is redundant to include leaves for both $(\mathbf{x}_p, \mathbf{x}_q)$ and $(\mathbf{x}_q, \mathbf{x}_p)$ in our tree. Instead, we can build separate trees to compute the diagonal and upper-triangular components of the sum, then reuse the upper-triangle result for the lower triangle.

Factorization of product distances. In general, computing the product distance δ will usually involve two calls to the underlying distance metric d ; these can often be reused. For example, when calculating both $\delta((\mathbf{x}_a, \mathbf{x}_b), (\mathbf{x}_c, \mathbf{x}_d))$ and $\delta((\mathbf{x}_a, \mathbf{x}_e), (\mathbf{x}_c, \mathbf{x}_d))$, we can reuse the value of $d(\mathbf{x}_a, \mathbf{x}_c)$ for both computations. This reduces the total number of calls to the distance function during tree construction from a worst-case n^4 (for all pairs of pairs of training points) to a maximum of n^2 , and in general much fewer if other optimizations such as sparsity are implemented as well.

Leaf binning at short distances. If all leaves below a node \mathbf{n} are within a kernel lengthscale of \mathbf{n} , we cut off the tree at \mathbf{n} and just compute the exact weighted sum over those leaves, avoiding the tree recursion.

4 MIXED LOCAL/GLOBAL GP REPRESENTATIONS

In this section, we extend the GP model (1) to include both a local and a global component $g(\mathbf{x}_i)$, i.e.,

$$y_i = h(\mathbf{x}_i) + \epsilon_i = f(\mathbf{x}_i) + g(\mathbf{x}_i) + \epsilon_i, \quad (9)$$

where f is modeled by a short-lengthscale/compactly-supported GP, and g is a global component of constant rank (i.e., not directly dependent on n). We will show how to

efficiently calculate posteriors from such models using a product tree.

Formally, we assume a GP of the form

$$h(X) \sim \mathcal{N}(\phi(X)^T \mathbf{b}, k_f(X) + \phi(X)^T B \phi(X)) \quad (10)$$

where $k_f(X)$ is a sparse matrix, B is an $m \times m$ matrix and \mathbf{b} an m -dimensional vector, and $\phi(X)$ computes an $n \times m$ feature representation where $m \ll n$. This “sparse+low rank” formulation includes a wide range of models capturing global and local structure. For example, we can express the “explicit basis functions” model from section 2.7 of Rasmussen and Williams (2006) by letting $\phi(X)$ denote the basis functions $H(X)$ and letting \mathbf{b}, B denote the mean and covariance of a Gaussian prior on their weights. Similarly, the CS+FIC model given by Vanhatalo and Vehtari (2008) may be represented² by taking $\phi(X) = K_{u,n}$, $B = K_{u,u}^{-1}$, $\mathbf{b} = \mathbf{0}$, and letting $k_f(X)$ absorb the diagonal term Λ . Other approximate GP models for global variation (e.g., Quiñero-Candela and Rasmussen, 2005; Snelson and Ghahramani, 2007; Rahimi and Recht, 2007; Vedaldi and Zisserman, 2010) can also be expressed in this form.

Given any model in the form of Eqn. (10), the posterior distribution $h(X^*) \sim \mathcal{N}(\mu'_*, \Sigma'_*)$ can be derived (Rasmussen and Williams, 2006) as

$$\mu'_* = \phi^{*T} \bar{\beta} + K^{*T} K_y^{-1} (\mathbf{y} - \phi^* \bar{\beta}) \quad (11)$$

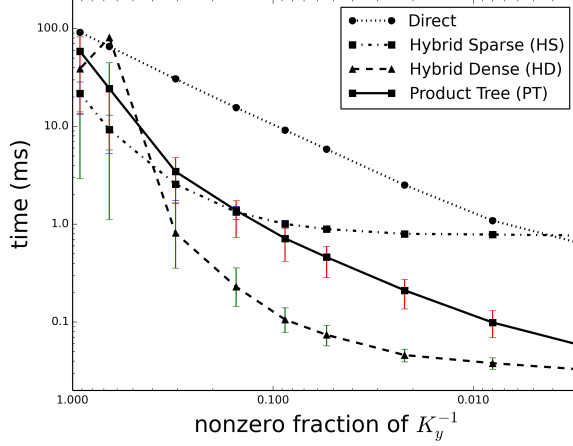
$$\Sigma'_* = K^{**} - K^{*T} K_y^{-1} K^* + R^T (B^{-1} + \phi K_y^{-1} \phi^T) R \quad (12)$$

where we let $\phi = \phi(X)$ and $\phi^* = \phi(X^*)$, and we have $\bar{\beta} = (B^{-1} + \phi K_y^{-1} \phi^T)^{-1} (\phi K_y^{-1} \mathbf{y} + B^{-1} \mathbf{b})$ and $R = \phi^* - \phi(X) K_y^{-1} K^*$. Section 2.7 of Rasmussen and Williams (2006) gives further details.

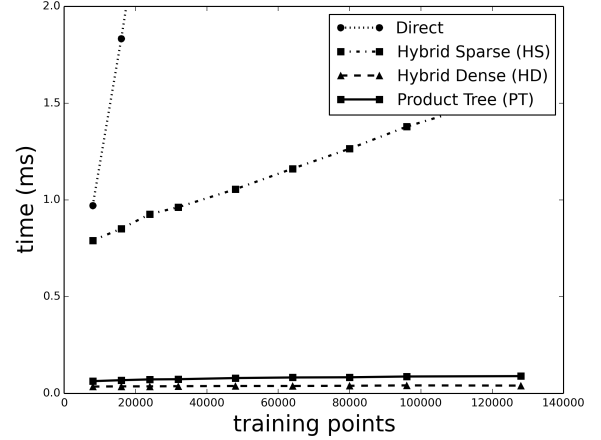
4.1 EFFICIENT OPERATIONS IN SPARSE+LOW RANK MODELS

Calculating the posterior given by (11, 12) is a straightforward extension of the standard case. The predictive

²Here the right side of each expression follows the notation of Vanhatalo and Vehtari.



(a) As a function of input density for a fixed-size (5000 points) training set, with error bars at the 10th to 90th percentiles.



(b) As a function of training set size with constant density $v = 5$.

Figure 3: Mean times to compute posterior variance on 2D synthetic data using a piecewise-polynomial $CS_{2,2}$ kernel.

mean (11) can be accommodated within the framework of Shen et al. (2006) using a tree representation of the vector $K_y^{-1}(\mathbf{y} - \phi^* \bar{\beta})$, then adding in the easily evaluated parametric component $\phi^* \bar{\beta}$. In the covariance (12) we can use a product tree to approximate $K^{*T} K_y^{-1} K^*$ as described above; of the remaining terms, $\bar{\beta}$ and $B^{-1} + \phi K_y^{-1} \phi^T$ can be precomputed at training time, and ϕ^* and K^{**} don't depend on the training set. This leaves $\phi K_y^{-1} K^*$ as the one remaining challenge; we note that this quantity can be computed efficiently using m applications per test point of the vector multiplication primitive from Shen et al. (2006), reusing the same tree structure to multiply each column of K^* by each row of ϕK_y^{-1} . Thus, the full posterior distribution at a test point can be calculated efficiently with no explicit dependence on n (i.e., with no direct access to the training points except through space-partitioning tree structures).

5 EVALUATION

We compare the use of a product tree (PT) for predictive variance calculation with several alternatives:

Direct: sparse matrix multiplication, using a sparse representation of K_y^{-1} and dense representation of \mathbf{k}_i^* .

Hybrid Sparse (HS): sparse matrix multiplication, using a sparse representation of \mathbf{k}_i^* constructed by querying a cover tree for all training points within distance r of the query point \mathbf{x}_i^* , where r is chosen such that $k(r')$ is negligible for $r' > r$, and then filling in only those entries of \mathbf{k}_i^* determined to be non-negligible. Since all our experiments involve kernels with compact support, we simply set r equal to the kernel lengthscale.

Hybrid Dense (HD):³ dense matrix multiplication, using only those entries of K_y^{-1} that correspond to training points within distance r of the query point. These training points are identified using a cover tree, as above, and entries of K_y^{-1} are retrieved from a hash table.

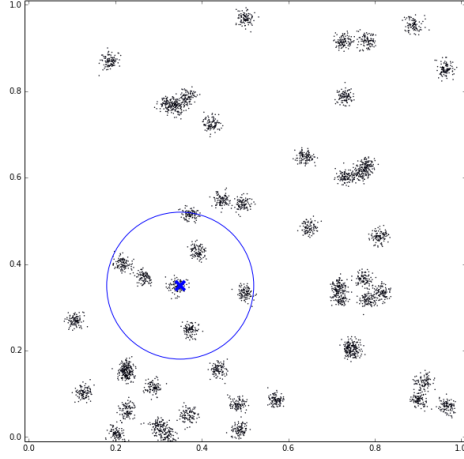
We do not show a comparison to the naïve dense matrix approach, since this is generally slower by orders of magnitude on the datasets we consider.

Our product tree implementation is a Python extension written in C++, based on the cover tree implementation of Beygelzimer et al. (2006) and implementing the optimizations from Section 3.3. In all experiments we set the approximation parameter ϵ_{rel} to ensure an approximation error of less than 0.1% of the exact variance. All sparse matrix multiplications are in CSR format using SciPy's sparse routines; we impose a sparsity threshold of 10^{-8} such that any entry less than the threshold is set to zero. Code to reproduce all experiments, along with the datasets, is included in the supplementary materials.

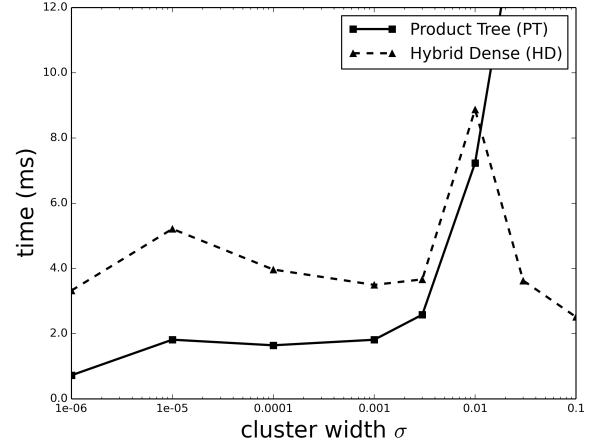
5.1 SYNTHETIC DATA

Figures 3a and 3b compare our methods on a simple two-dimensional synthetic data set, consisting of points sampled uniformly at random from the unit square. We train a GP on n such points and then measure the mean time per point to compute the predictive variance at 1000 random test points. The GP uses a piecewise-polynomial $CS_{2,2}$ kernel with observation noise $\sigma_n^2 = 1.0$ and lengthscale $\ell = \sqrt{v\pi/n}$, where v is a parameter indicating the average number of training points within a one-lengthscale ball of

³We are grateful to Iain Murray for suggesting this approach for comparison.



(a) Clumpy data with $\sigma = 0.01$, with a ball of covariance lengthscale $\frac{1}{\sqrt{10\pi}}$ overlaid.



(b) Mean times to compute posterior variance on data of varying clumpiness, using a piecewise-polynomial $CS_{2,2}$ kernel.

Figure 4: HD versus PT on clumpy data.

a random query point.

We see from Figure 3a that the hybrid dense method performs best when the kernel lengthscale is extremely short, followed by the product tree; in the most extreme cases these methods outperform the alternatives by an order of magnitude. However, performance degrades as the kernel lengthscale increases.

Figure 3b examines the scaling behavior of the algorithms in a relatively sparse setting, $v = 5$, chosen to allow the tractable inversion of large kernel matrices. Here we see that the direct calculation scales quite steeply (though linearly: the runtime at $n = 160000$ is approximately 15ms) with training size due to the need to explicitly compute all n entries of \mathbf{k}_i^* . The hybrid sparse calculation avoids this bottleneck and is significantly more efficient, but its scaling is ultimately also linear with n , an inherent limitation of sparse matrix multiplication (Bank and Douglas, 1993) since it does not have access to the geometry of the data. By contrast, in this sparse setting the hybrid dense and product tree approaches remain efficient even for very large datasets, with a small constant-factor advantage for the hybrid dense method.

5.2 CLUMPINESS

The strong performance in the previous experiments of the hybrid dense method, relative to the product tree, is due to the uniformity of the training data. With no natural clusters, the only available optimization is to discard faraway points. The product tree would be expected to perform better when the data are ‘clumpy’, allowing it to merge kernel evaluations from nearby points. This is explored in Figure 4b, which compares the two methods on a synthetic dataset of 5000 points, sampled from a mixture of 50 Gaus-

sians each with covariance $\sigma^2 \mathcal{I}$ for varying clumpiness σ . As expected, the product tree is fastest when the data are tightly clustered and many points can be merged. Figure 4a shows an example of a dataset at $\sigma=0.01$, the approximate ‘crossover’ point at which the lower constant factor of the hybrid dense method begins to outweigh the advantages of the product tree.

5.3 REAL DATA

We evaluate the performance of our methods on the following datasets, shown in Table 2:

seismic: 20000 travel-time residuals between observed P-wave travel times to the seismic array in Alice Springs, Australia, and the times predicted by a one-dimensional IASPEI91 model (Kennett and Engdahl, 1991), indexed by latitude/longitude and depth of the source event.

snow: 20000 observations of water content of California snow pack recorded daily at 128 stations from November 1, 2011 to June 1, 2012, indexed by date, latitude/longitude and elevation. Collected from <http://cdec.water.ca.gov/queryCSV.html>.

precip: shown in Figure 1, total annual precipitation recorded in 1995 by each of 5775 stations in the continental US, indexed by latitude, longitude, and elevation (Vanhatalo and Vehtari, 2008).

tco: Total column ozone as recorded over the Earth’s surface by the NIMBUS-7/TOMS satellite on October 1, 1998 (Park et al., 2011). Our experiments use a random sample of 20000 from the full 48331 measurements.

housing: Data from the 1990 California census, as used by Shen et al. (2006). We predict the median income of each block group as a function of median age and median house

value.

Table 3 compares, for each dataset, the performance of an SE kernel to that of a piecewise-polynomial, compactly-supported kernel $CS_{D,2}$ with a hand-selected number of FIC inducing points capturing global variation. The goal here is to show that the CS+FIC models, which are well-suited for fast tree-based calculations at test time, are a reasonable modeling choice even for purely predictive reasons. Model quality is measured by the Standardized Mean Squared Error (SMSE), i.e., the squared error divided by the squared error of the trivial predictor that just predicts the mean of the training set, and Mean Standardized Log Loss (MSLL), obtained by averaging $-\log p(y_i^*|x_i^*, X, \mathbf{y})$ over the test set and subtracting the same score for a trivial model which always predicts the mean and variance of the training set. Hyperparameters for each model were obtained by maximizing the marginal likelihood over a random subset of 5000 training points using a truncated Newton method; a prior was used to encourage short lengthscales for the CS components of the CS+FIC models. Note that, for each of the datasets considered in this paper, the CS+FIC model provides better posterior probability estimates (lower MSLL) than an SE model.⁴

In Table 4 we show, for each method and dataset, the mean and standard deviation of posterior variance computation time evaluated over the test set. Here the **HS**, **HD**, and **PT** methods use the tree-optimized FIC calculations from Section 4.1, while the **Direct** and **HS_N** methods use a naive FIC calculation; the latter is included explicitly for comparison with the tree-optimized version. Interestingly, the hybrid dense calculation is quickest in every case, sometimes tied by the product tree, suggesting that these real datasets do not possess the degree of clumpiness necessary for the product tree to dominate (Table 5 directly compares the number of terms used by the two method, showing that the product tree succeeds in merging a significant number of points only on the housing dataset). Both the product tree and the hybrid dense method are generally faster than the hybrid sparse method, with the exception of the US precipitation data in which the relative fullness of the inverse

⁴The SE model may still be superior in many cases, of course. For example, we experimented with the well-known SARCOS inverse kinematics dataset but were unable to find a CS+FIC model that was competitive with the SE baseline.

	d	n_{train}	n_{test}	t_{build}
seismic	3	16000	4000	4.9s
snow	4	15000	5000	4.2s
precip	3	5000	775	23.0s
tco	2	15000	5000	2.2s
housing	2	18000	2000	3.3s

Table 2: Datasets, with product tree construction times.

	model	K_y^{-1} %	SMSE	MSLL
seismic	CS+FIC (20)	0.6%	0.82	-0.20
	SE	33.9%	0.84	-0.11
snow	CS+FIC (20)	0.8%	0.0030	-2.91
	SE	36.3%	0.0097	-2.31
precip	CS+FIC (20)	45.3%	0.129	-1.17
	SE	50.2%	0.125	-1.06
tco	CS+FIC (90)	0.4%	0.041	-1.63
	SE	34.6%	0.054	-1.45
housing	CS+FIC (20)	0.5%	0.83	-0.18
	SE	100%	0.80	-0.086

Table 3: Predictive performance of compactly-supported and squared-exponential kernels on the test datasets. Smaller is better for both SMSE and MSLL.

kernel matrix for that model (Table 3) greatly increases the size of the product tree. Comparing **HS** to **HS_N**, we see that the tree-optimized FIC calculation provides significant speedups on all datasets except for the precipitation data, with an especially significant speedup for the **tco** data which uses 90 inducing points.

6 CONCLUSION AND FUTURE WORK

This paper introduces the *product tree*, a method for efficient adaptive calculation of GP covariances using a multiresolution clustering of pairs of training points. We empirically evaluate the performance of several such methods and find that a simple heuristic that discards faraway training points (the hybrid dense method described above) may yield the best performance on many real datasets. This follows Murray (2009), who found that tree-based methods are often unable to effectively merge points, though we do identify a regime of clustered data in which the product tree has an advantage. Other contributions of this paper include a cutoff rule with provable error bounds, applicable to both mean and covariance calculations on trees, and a description of efficient calculation in GP models incorporating both sparse and low-rank components, showing how such models can model global-scale variation while maintaining the efficiency of short-lengthscale GPs.

A limitation of all of the approaches considered in this paper is the need to invert the kernel matrix during training; this can be difficult for large problems. One avenue for future work could be an iterative factorization of K_y analogous to the CG training performed by MVM methods (Shen et al., 2006; Gray, 2004; Morariu et al., 2008).

Although our work has been focused primarily on low-dimensional applications, the use of cover trees instead of k -d trees ought to enable an extension to higher dimensions. We are not aware of previous work applying tree-based regression algorithms to high-dimensional data, but

	Direct (ms)	HS _N (ms)	HS (ms)	HD (ms)	PT (ms)
seismic	13.1 ± 1.1	4.5 ± 0.9	2.0 ± 1.6	1.5 ± 2.7	1.6 ± 2.4
precip	45.0 ± 2.0	5.0 ± 1.5	5.7 ± 2.3	3.0 ± 1.0	7.3 ± 3.4
snow	13.0 ± 1.0	4.3 ± 0.6	1.6 ± 0.4	0.9 ± 0.5	1.0 ± 0.5
tco	19.4 ± 4.9	17.4 ± 6.4	2.7 ± 1.2	1.7 ± 0.2	1.7 ± 0.5
housing	12.2 ± 1.0	4.9 ± 0.8	1.5 ± 0.3	0.8 ± 0.3	0.8 ± 0.3

Table 4: Time to compute posterior variance of a CS+FIC model at points from the test set: mean ± standard deviation.

	HD	PT
seismic	6912	6507
precip	21210	22849
snow	1394	1374
tco	169	177
housing	1561	670

Table 5: Mean number of nonzero terms in the approximate weighted sums computed at test points.

as high-dimensional covariance matrices are often sparse, this may be a natural fit. For high-dimensional data that do not lie on a low-dimensional manifold, other nearest-neighbor techniques such as locality-sensitive hashing (Andoni and Indyk, 2008) may have superior properties to tree structures; the adaptation of such techniques to GP regression is an interesting open problem.

Acknowledgements

The authors are grateful to the anonymous reviewers for their helpful feedback, and to Iain Murray for suggesting the hybrid dense method. This work was supported by DTRA grant #HDTRA-11110026.

References

- Andoni, A. and Indyk, P. (2008). Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Communications of the ACM*, 51(1):117–122.
- Bank, R. E. and Douglas, C. C. (1993). Sparse matrix multiplication package (SMMP). *Advances in Computational Mathematics*, 1(1):127–137.
- Beygelzimer, A., Kakade, S., and Langford, J. (2006). Cover trees for nearest neighbor. In *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, pages 97–104.
- Bickel, P. J. and Lindner, M. (2012). Approximating the inverse of banded matrices by banded matrices with applications to probability and statistics. *SIAM Journal on Probability Theory and Applications*, 56:1–20.
- Chalupka, K., Williams, C. K., and Murray, I. (2013). A framework for evaluating approximation methods for Gaussian process regression. *Journal of Machine Learning Research*, 14:333–350.
- Friedman, J. H., Bentley, J. L., and Finkel, R. A. (1977). An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software (TOMS)*, 3(3):209–226.
- Gray, A. (2004). Fast kernel matrix-vector multiplication with application to Gaussian process learning. Technical Report CMU-CS-04-110, School of Computer Science, Carnegie Mellon University.
- Gray, A. G. and Moore, A. W. (2001). N-body problems in statistical learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 521–527.
- Kennett, B. N. and Engdahl, E. (1991). Traveltimes for global earthquake location and phase identification. *Geophysical Journal International*, 105(2):429–465.
- Moore, A. W., Schneider, J., and Deng, K. (1997). Efficient locally weighted polynomial regression predictions. In *Proceedings of the 14th International Conference on Machine Learning (ICML)*.
- Morariu, V., Srinivasan, B. V., Raykar, V. C., Duraiswami, R., and Davis, L. (2008). Automatic online tuning for fast Gaussian summation. *Advances in Neural Information Processing Systems (NIPS)*, 21:1113–1120.
- Murray, I. (2009). Gaussian processes and fast matrix-vector multiplies. In *Numerical Mathematics in Machine Learning workshop at the 26th International Conference on Machine Learning (ICML 2009)*.
- Park, C., Huang, J. Z., and Ding, Y. (2011). Domain decomposition approach for fast gaussian process regression of large spatial data sets. *The Journal of Machine Learning Research*, 12:1697–1728.
- Quiñonero-Candela, J. and Rasmussen, C. E. (2005). A unifying view of sparse approximate Gaussian process regression. *The Journal of Machine Learning Research*, 6:1939–1959.
- Rahimi, A. and Recht, B. (2007). Random features for large-scale kernel machines. *Advances in Neural Information Processing Systems (NIPS)*, 20:1177–1184.

- Rasmussen, C. and Williams, C. (2006). *Gaussian Processes for Machine Learning*. MIT Press.
- Shen, Y., Ng, A., and Seeger, M. (2006). Fast Gaussian process regression using kd-trees. In *Advances in Neural Information Processing Systems (NIPS)*, volume 18, page 1225.
- Snelson, E. and Ghahramani, Z. (2006). Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems (NIPS)*.
- Snelson, E. and Ghahramani, Z. (2007). Local and global sparse Gaussian process approximations. In *Artificial Intelligence and Statistics (AISTATS)*, volume 11.
- Uhlmann, J. K. (1991). Satisfying general proximity / similarity queries with metric trees. *Information Processing Letters*, 40(4):175 – 179.
- Vanhatalo, J. and Vehtari, A. (2008). Modelling local and global phenomena with sparse Gaussian processes. In *Proceedings of Uncertainty in Artificial Intelligence (UAI)*.
- Vedaldi, A. and Zisserman, A. (2010). Efficient additive kernels via explicit feature maps. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3539–3546. IEEE.

Asymptotically Exact, Embarrassingly Parallel MCMC

Willie Neiswanger

Machine Learning Department
Carnegie Mellon University
willie@cs.cmu.edu

Chong Wang

Voleon Capital Management
chongw@cs.princeton.edu

Eric P. Xing

School of Computer Science
Carnegie Mellon University
epxing@cs.cmu.edu

Abstract

Communication costs, resulting from synchronization requirements during learning, can greatly slow down many parallel machine learning algorithms. In this paper, we present a parallel Markov chain Monte Carlo (MCMC) algorithm in which subsets of data are processed independently, with very little communication. First, we arbitrarily partition data onto multiple machines. Then, on each machine, any classical MCMC method (e.g., Gibbs sampling) may be used to draw samples from a posterior distribution given the data subset. Finally, the samples from each machine are combined to form samples from the full posterior. This embarrassingly parallel algorithm allows each machine to act independently on a subset of the data (without communication) until the final combination stage. We prove that our algorithm generates asymptotically exact samples and empirically demonstrate its ability to parallelize burn-in and sampling in several models.

1 Introduction

Markov chain Monte Carlo (MCMC) methods are popular tools for performing approximate Bayesian inference via posterior sampling. One major benefit of these techniques is that they guarantee asymptotically exact recovery of the posterior distribution as the number of posterior samples grows. However, MCMC methods may take a prohibitively long time, since for N data points, most methods must perform $O(N)$ operations to draw a sample. Furthermore, MCMC methods might require a large number of “burn-in” steps before beginning to generate representative samples. Further complicating matters is the issue that, for many big data applications, it is necessary to store and process

data on multiple machines, and so MCMC must be adapted to run in these data-distributed settings.

Researchers currently tackle these problems independently, in two primary ways. To speed up sampling, multiple independent chains of MCMC can be run in parallel [20, 11, 13]; however, each chain is still run on the entire dataset, and there is no speed-up of the burn-in process (as each chain must still complete the full burn-in before generating samples). To run MCMC when data is partitioned among multiple machines, each machine can perform computation that involves a subset of the data and exchange information at each iteration to draw a sample [10, 14, 18]; however, this requires a significant amount of communication between machines, which can greatly increase computation time when machines wait for external information [1, 7].

We aim to develop a procedure to tackle both problems simultaneously, to allow for quicker burn-in and sampling in settings where data are partitioned among machines. To accomplish this, we propose the following: on each machine, run MCMC on only a subset of the data (independently, without communication between machines), and then combine the samples from each machine to algorithmically construct samples from the full-data posterior distribution. We’d like our procedure to satisfy the following four criteria:

1. Each machine only has access to a portion of the data.
2. Each machine performs MCMC independently, without communicating (i.e. “embarrassingly parallel”).
3. Each machine can use any type of MCMC to generate samples.
4. The combination procedure yields provably asymptotically exact samples from the full-data posterior.

The third criterion allows existing MCMC algorithms or software packages to be run directly on subsets of the data—the combination procedure then acts as a post-processing step to transform the samples to the correct distribution. Note that this procedure is particularly

suitable for use in a MapReduce [4] framework. Also note that, unlike current strategies, this procedure does not involve multiple “duplicate” chains (as each chain uses a different portion of the data and samples from a different posterior distribution), nor does it involve parallelizing a single chain (as there are multiple chains operating independently). We will show how this allows our method to, in fact, parallelize and greatly reduce the time required for burn-in.

In this paper we will (1) introduce and define the *subposterior* density—a modified posterior given a subset of the data—which will be used heavily, (2) present methods for the embarrassingly parallel MCMC and combination procedure, (3) prove theoretical guarantees about the samples generated from our algorithm, (4) describe the current scope of the presented method (i.e. where and when it can be applied), and (5) show empirical results demonstrating that we can achieve speed-ups for burn-in and sampling while meeting the above four criteria.

2 Embarrassingly Parallel MCMC

The basic idea behind our method is to partition a set of N i.i.d. data points $x^N = \{x_1, \dots, x_N\}$ into M subsets, sample from the *subposterior*—the posterior given a data subset with an underweighted prior—in parallel, and then combine the resulting samples to form samples from the full-data posterior $p(\theta|x^N)$, where $\theta \in \mathbb{R}^d$ and $p(\theta|x^N) \propto p(\theta)p(x^N|\theta) = p(\theta)\prod_{i=1}^N p(x_i|\theta)$.

More formally, given data x^N partitioned into M subsets $\{x^{n_1}, \dots, x^{n_M}\}$, the procedure is:

1. For $m = 1, \dots, M$ (in parallel):
Sample from the subposterior p_m , where
$$p_m(\theta) \propto p(\theta)^{\frac{1}{M}} p(x^{n_m}|\theta). \quad (1)$$
2. Combine the subposterior samples to produce samples from an estimate of the subposterior density product $p_1 \cdots p_M$, which is proportional to the full-data posterior, i.e. $p_1 \cdots p_M(\theta) \propto p(\theta|x^N)$.

We want to emphasize that we do not need to iterate over these steps and the combination stage (step 3) is the only step that requires communication between machines. Also note that sampling from each subposterior (step 2) can typically be done in the same way as one would sample from the full-data posterior. For example, when using the Metropolis-Hastings algorithm, one would compute the likelihood ratio as $\frac{p(\theta^*)^{\frac{1}{M}} p(x^{n_m}|\theta^*)}{p(\theta)^{\frac{1}{M}} p(x^{n_m}|\theta)}$ instead of $\frac{p(\theta^*)p(x^N|\theta^*)}{p(\theta)p(x^N|\theta)}$, where θ^* is the proposed move. In the next section, we show how the combination stage (step 3) is carried out to generate samples from the full-data posterior using the subposterior samples.

3 Combining Subposterior Samples

Our general idea is to combine the subposterior samples in such a way that we are implicitly sampling from an estimate of the subposterior density product function $\widehat{p_1 \cdots p_M}(\theta)$. If our density product estimator is consistent, then we can show that we are drawing asymptotically exact samples from the full posterior. Further, by studying the estimator error rate, we can explicitly analyze how quickly the distribution from which we are drawing samples is converging to the true posterior (and thus compare different combination algorithms).

In the following three sections we present procedures that yield samples from different estimates of the density product. Our first example is based on a simple parametric estimator motivated by the Bernstein-von Mises theorem [12]; this procedure generates approximate (asymptotically biased) samples from the full posterior. Our second example is based on a nonparametric estimator, and produces asymptotically exact samples from the full posterior. Our third example is based on a semiparametric estimator, which combines beneficial aspects from the previous two estimators while also generating asymptotically exact samples.

3.1 Approximate posterior sampling with a parametric estimator

The first method for forming samples from the full posterior given subposterior samples involves using an approximation based on the Bernstein-von Mises (Bayesian central limit) theorem, an important result in Bayesian asymptotic theory. Assuming that a unique, true data-generating model exists and is denoted θ_0 , this theorem states that the posterior tends to a normal distribution concentrated around θ_0 as the number of observations grows. In particular, under suitable regularity conditions, the posterior $P(\theta|x^N)$ is well approximated by $\mathcal{N}_d(\theta_0, F_N^{-1})$ (where F_N is the fisher information of the data) when N is large [12]. Since we aim to perform posterior sampling when the number of observations is large, a normal parametric form often serves as a good posterior approximation. A similar approximation was used in [2] in order to facilitate fast, approximately correct sampling. We therefore estimate each subposterior density with $\hat{p}_m(\theta) = \mathcal{N}_d(\theta|\hat{\mu}_m, \hat{\Sigma}_m)$ where $\hat{\mu}_m$ and $\hat{\Sigma}_m$ are the sample mean and covariance, respectively, of the subposterior samples. The product of the M subposterior densities will be proportional to a Gaussian pdf, and our estimate of the density product function $p_1 \cdots p_M(\theta) \propto p(\theta|x^N)$ is

$$\widehat{p_1 \cdots p_M}(\theta) = \hat{p}_1 \cdots \hat{p}_M(\theta) \propto \mathcal{N}_d(\theta|\hat{\mu}_M, \hat{\Sigma}_M),$$

where the parameters of this distribution are

$$\hat{\Sigma}_M = \left(\sum_{m=1}^M \hat{\Sigma}_m^{-1} \right)^{-1} \quad (2)$$

$$\hat{\mu}_M = \hat{\Sigma}_M \left(\sum_{m=1}^M \hat{\Sigma}_m^{-1} \hat{\mu}_m \right). \quad (3)$$

These parameters can be computed quickly and, if desired, online (as new subposterior samples arrive).

3.2 Asymptotically exact posterior sampling with nonparametric density product estimation

In the previous method we made a parametric assumption based on the Bernstein-von Mises theorem, which allows us to generate approximate samples from the full posterior. Although this parametric estimate has quick convergence, it generates asymptotically biased samples, especially in cases where the posterior is particularly non-Gaussian. In this section, we develop a procedure that implicitly samples from the product of nonparametric density estimates, which allows us to produce asymptotically exact samples from the full posterior. By constructing a consistent density product estimator from which we can generate samples, we ensure that the distribution from which we are sampling converges to the full posterior.

Given T samples¹ $\{\theta_{t_m}^m\}_{t_m=1}^T$ from a subposterior p_m , we can write the kernel density estimator $\hat{p}_m(\theta)$ as,

$$\begin{aligned} \hat{p}_m(\theta) &= \frac{1}{T} \sum_{t_m=1}^T \frac{1}{h^d} K\left(\frac{\|\theta - \theta_{t_m}^m\|}{h}\right) \\ &= \frac{1}{T} \sum_{t_m=1}^T \mathcal{N}_d(\theta | \theta_{t_m}^m, h^2 I_d), \end{aligned}$$

where we have used a Gaussian kernel with bandwidth parameter h . After we have obtained the kernel density estimator $\hat{p}_m(\theta)$ for M subposteriors, we define our nonparametric density product estimator for the full posterior as

$$\begin{aligned} \widehat{p_1 \cdots p_M}(\theta) &= \hat{p}_1 \cdots \hat{p}_M(\theta) \\ &= \frac{1}{T^M} \prod_{m=1}^M \sum_{t_m=1}^T \mathcal{N}_d(\theta | \theta_{t_m}^m, h^2 I_d) \\ &\propto \sum_{t_1=1}^T \cdots \sum_{t_M=1}^T w_{t.} \mathcal{N}_d\left(\theta \middle| \bar{\theta}_{t.}, \frac{h^2}{M} I_d\right). \quad (4) \end{aligned}$$

¹For ease of description, we assume each machine generates the same number of samples, T . In practice, they do not have to be the same.

This estimate is the probability density function (pdf) of a mixture of T^M Gaussians with *unnormalized* mixture weights $w_{t.}$. Here, we use $t. = \{t_1, \dots, t_M\}$ to denote the set of indices for the M samples $\{\theta_{t_1}^1, \dots, \theta_{t_M}^M\}$ (each from a separate machine) associated with a given mixture component, and we define

$$\bar{\theta}_{t.} = \frac{1}{M} \sum_{m=1}^M \theta_{t_m}^m \quad (5)$$

$$w_{t.} = \prod_{m=1}^M \mathcal{N}_d(\theta_{t_m}^m | \bar{\theta}_{t.}, h^2 I_d). \quad (6)$$

Although there are T^M possible mixture components, we can efficiently generate samples from this mixture by first sampling a mixture component (based on its unnormalized component weight $w_{t.}$) and then sampling from this (Gaussian) component. In order to sample mixture components, we use an independent Metropolis within Gibbs (IMG) sampler. This is a form of MCMC, where at each step in the Markov chain, a single dimension of the current state is proposed (i.e. sampled) independently of its current value (while keeping the other dimensions fixed) and then is accepted or rejected. In our case, at each step, a new mixture component is proposed by redrawing one of the M current sample indices $t_m \in t.$ associated with the component uniformly and then accepting or rejecting the resulting proposed component based on its mixture weight. We give the IMG algorithm for combining subposterior samples in Algorithm 1.²

In certain situations, Algorithm 1 may have a low acceptance rate and therefore may mix slowly. One way to remedy this is to perform the IMG combination algorithm multiple times, by first applying it to groups of $\tilde{M} < M$ subposteriors and then applying the algorithm again to the output samples from each initial application. For example, one could begin by applying the algorithm to all $\frac{M}{2}$ pairs (leaving one subposterior alone if M is odd), then repeating this process—forming pairs and applying the combination algorithm to pairs only—until there is only one set of samples remaining, which are samples from the density product estimate.

3.3 Asymptotically exact posterior sampling with semiparametric density product estimation

Our first example made use of a parametric estimator, which has quick convergence, but may be asymptotically biased, while our second example made use of

²Again for simplicity, we assume that we generate T samples to represent the full posterior, where T is the number of subposterior samples from each machine.

Algorithm 1 Asymptotically Exact Sampling via Non-parametric Density Product Estimation

Input: Subposterior samples: $\{\theta_{t_1}^1\}_{t_1=1}^T \sim p_1(\theta), \dots, \{\theta_{t_M}^M\}_{t_M=1}^T \sim p_M(\theta)$

Output: Posterior samples (asymptotically, as $T \rightarrow \infty$): $\{\theta_i\}_{i=1}^T \sim p_1 \cdots p_M(\theta) \propto p(\theta|x^N)$

```

1: Draw  $t \cdot = \{t_1, \dots, t_M\} \stackrel{\text{iid}}{\sim} \text{Unif}(\{1, \dots, T\})$ 
2: for  $i = 1$  to  $T$  do
3:   Set  $h \leftarrow i^{-1/(4+d)}$ 
4:   for  $m = 1$  to  $M$  do
5:     Set  $c \leftarrow t \cdot$ 
6:     Draw  $c_m \sim \text{Unif}(\{1, \dots, T\})$ 
7:     Draw  $u \sim \text{Unif}([0, 1])$ 
8:     if  $u < w_{c \cdot}/w_t$  then
9:       Set  $t \cdot \leftarrow c \cdot$ 
10:    end if
11:  end for
12:  Draw  $\theta_i \sim \mathcal{N}_d(\bar{\theta}_t, \frac{h^2}{M} I_d)$ 
13: end for
```

a nonparametric estimator, which is asymptotically exact, but may converge slowly when the number of dimensions is large. In this example, we implicitly sample from a semiparametric density product estimate, which allows us to leverage the fact that the full posterior has a near-Gaussian form when the number of observations is large, while still providing an asymptotically unbiased estimate of the posterior density, as the number of subposterior samples $T \rightarrow \infty$.

We make use of a semiparametric density estimator for p_m that consists of the product of a parametric estimator $\hat{f}_m(\theta)$ (in our case $\mathcal{N}_d(\theta|\hat{\mu}_m, \hat{\Sigma}_m)$ as above) and a nonparametric estimator $\hat{r}(\theta)$ of the correction function $r(\theta) = p_m(\theta)/\hat{f}_m(\theta)$ [6]. This estimator gives a near-Gaussian estimate when the number of samples is small, and converges to the true density as the number of samples grows. Given T samples $\{\theta_{t_m}^m\}_{t_m=1}^T$ from a subposterior p_m , we can write the estimator as

$$\begin{aligned}
\hat{p}_m(\theta) &= \hat{f}_m(\theta) \hat{r}(\theta) \\
&= \frac{1}{T} \sum_{t_m=1}^T \frac{1}{h^d} K\left(\frac{\|\theta - \theta_{t_m}^m\|}{h}\right) \frac{\hat{f}_m(\theta)}{\hat{f}_m(\theta_{t_m}^m)} \\
&= \frac{1}{T} \sum_{t_m=1}^T \frac{\mathcal{N}_d(\theta|\theta_{t_m}^m, h^2 I_d) \mathcal{N}_d(\theta|\hat{\mu}_m, \hat{\Sigma}_m)}{\mathcal{N}_d(\theta_{t_m}^m|\hat{\mu}_m, \hat{\Sigma}_m)},
\end{aligned}$$

where we have used a Gaussian kernel with bandwidth parameter h for the nonparametric component of this estimator. Therefore, we define our semiparametric

density product estimator to be

$$\begin{aligned}
\widehat{p_1 \cdots p_M}(\theta) &= \hat{p}_1 \cdots \hat{p}_M(\theta) \\
&= \frac{1}{T^M} \prod_{m=1}^M \sum_{t_m=1}^T \frac{\mathcal{N}_d(\theta|\theta_{t_m}^m, h I_d) \mathcal{N}_d(\theta|\hat{\mu}_m, \hat{\Sigma}_m)}{h^d \mathcal{N}_d(\theta_{t_m}^m|\hat{\mu}_m, \hat{\Sigma}_m)} \\
&\propto \sum_{t_1=1}^T \cdots \sum_{t_M=1}^T W_{t \cdot} \mathcal{N}_d(\theta|\mu_{t \cdot}, \Sigma_{t \cdot}).
\end{aligned}$$

This estimate is proportional to the pdf of a mixture of T^M Gaussians with unnormalized mixture weights,

$$W_{t \cdot} = \frac{w_{t \cdot} \mathcal{N}_d\left(\bar{\theta}_t, |\hat{\mu}_M, \hat{\Sigma}_M + \frac{h}{M} I_d\right)}{\prod_{m=1}^M \mathcal{N}_d(\theta_{t_m}^m|\hat{\mu}_m, \hat{\Sigma}_m)},$$

where $\bar{\theta}_t$ and w_t are given in Eqs. 5 and 6. We can write the parameters of a given mixture component $\mathcal{N}_d(\theta|\mu_{t \cdot}, \Sigma_{t \cdot})$ as

$$\begin{aligned}
\Sigma_{t \cdot} &= \left(\frac{M}{h} I_d + \hat{\Sigma}_M^{-1} \right)^{-1}, \\
\mu_{t \cdot} &= \Sigma_{t \cdot} \left(\frac{M}{h} I_d \bar{\theta}_t + \hat{\Sigma}_M^{-1} \hat{\mu}_M \right),
\end{aligned}$$

where $\hat{\mu}_M$ and $\hat{\Sigma}_M$ are given by Eq. 2 and 3. We can sample from this semiparametric estimate using the IMG procedure outlined in Algorithm 1, replacing the component weights w_t with W_t and the component parameters $\bar{\theta}_t$ and $\frac{h}{M} I_d$ with $\mu_{t \cdot}$ and $\Sigma_{t \cdot}$.

We also have a second semiparametric procedure that may give higher acceptance rates in the IMG algorithm. We follow the above semiparametric procedure, where each component is a normal distribution with parameters $\mu_{t \cdot}$ and $\Sigma_{t \cdot}$, but we use the nonparametric component weights w_t instead of W_t . This procedure is also asymptotically exact, since the semiparametric component parameters $\mu_{t \cdot}$ and $\Sigma_{t \cdot}$ approach the nonparametric component parameters $\bar{\theta}_t$ and $\frac{h}{M} I_d$ as $h \rightarrow 0$, and thus this procedure tends to the nonparametric procedure given in Algorithm 1.

4 Method Complexity

Given M data subsets, to produce T samples in d dimensions with the nonparametric or semiparametric asymptotically exact procedures (Algorithm 1) requires $O(dTM^2)$ operations. The variation on this algorithm that performs this procedure $M-1$ times on pairs of subposteriors (to increase the acceptance rate; detailed in Section 3.2) instead requires only $O(dTM)$ operations.

We have presented our method as a two step procedure, where first parallel MCMC is run to completion, and

then the combination algorithm is applied to the M sets of samples. We can instead perform an online version of our algorithm: as each machine generates a sample, it immediately sends it to a master machine, which combines the incoming samples³ and performs the accept or reject step (Algorithm 1, lines 3-12). This allows the parallel MCMC phase and the combination phase to be performed in parallel, and does not require transferring large volumes of data, as only a single sample is ever transferred at a time.

The total communication required by our method is transferring $O(dTM)$ scalars (T samples from each of M machines), and as stated above, this can be done online as MCMC is being carried out. Further, the communication is unidirectional, and each machine does not pause and wait for any information from other machines during the parallel sampling procedure.

5 Theoretical Results

Our second and third procedures aim to draw asymptotically exact samples by sampling from (fully or partially) nonparametric estimates of the density product. We prove the asymptotic correctness of our estimators, and bound their rate of convergence. This will ensure that we are generating asymptotically correct samples from the full posterior as the number of samples T from each subposterior grows.

5.1 Density product estimate convergence and risk analysis

To prove (mean-square) consistency of our estimator, we give a bound on the mean-squared error (MSE), and show that it tends to zero as we increase the number of samples drawn from each subposterior. To prove this, we first bound the bias and variance of the estimator. The following proofs make use of similar bounds on the bias and variance of the nonparametric and semiparametric density estimators, and therefore the theory applies to both the nonparametric and semiparametric density product estimators.

Throughout this analysis, we assume that we have T samples $\{\theta_{tm}^m\}_{t=1}^T \subset \mathcal{X} \subset \mathbb{R}^d$ from each subposterior ($m = 1, \dots, M$), and that $h \in \mathbb{R}_+$ denotes the bandwidth of the nonparametric density product estimator (which is annealed to zero as $T \rightarrow \infty$ in Algorithm 1). Let Hölder class $\Sigma(\beta, L)$ on \mathcal{X} be defined as the set of all $\ell = \lfloor \beta \rfloor$ times differentiable functions $f : \mathcal{X} \rightarrow \mathbb{R}$ whose derivative $f^{(\ell)}$ satisfies

$$|f^{(\ell)}(\theta) - f^{(\ell)}(\theta')| \leq L |\theta - \theta'|^{\beta-\ell} \quad \text{for all } \theta, \theta' \in \mathcal{X}.$$

³For the semiparametric method, this will involve an online update of mean and variance Gaussian parameters.

We also define the class of densities $\mathcal{P}(\beta, L)$ to be

$$\mathcal{P}(\beta, L) = \left\{ p \in \Sigma(\beta, L) \mid p \geq 0, \int p(\theta) d\theta = 1 \right\}.$$

We also assume that all subposterior densities p_m are bounded, i.e. that there exists some $b > 0$ such that $p_m(\theta) \leq b$ for all $\theta \in \mathbb{R}^d$ and $m \in \{1, \dots, M\}$.

First, we bound the bias of our estimator. This shows that the bias tends to zero as the bandwidth shrinks.

Lemma 5.1. *The bias of the estimator $\widehat{p_1 \cdots p_M}(\theta)$ satisfies*

$$\sup_{p_1, \dots, p_M \in \mathcal{P}(\beta, L)} |\mathbb{E} [\widehat{p_1 \cdots p_M}(\theta)] - p_1 \cdots p_M(\theta)| \leq \sum_{m=1}^M c_m h^{m\beta}$$

for some $c_1, \dots, c_M > 0$.

Proof. For all $p_1, \dots, p_M \in \mathcal{P}(\beta, L)$,

$$\begin{aligned} |\mathbb{E} [\widehat{p_1 \cdots p_M}] - p_1 \cdots p_M| &= |\mathbb{E} [\widehat{p_1} \cdots \widehat{p_M}] - p_1 \cdots p_M| \\ &= |\mathbb{E} [\widehat{p_1}] \cdots \mathbb{E} [\widehat{p_M}] - p_1 \cdots p_M| \\ &\leq |(p_1 + \tilde{c}_1 h^\beta) \cdots (p_M + \tilde{c}_M h^\beta) - p_1 \cdots p_M| \\ &\leq |c_1 h^\beta + \dots + c_M h^{M\beta}| \\ &\leq |c_1 h^\beta| + \dots + |c_M h^{M\beta}| \\ &= \sum_{m=1}^M c_m h^{m\beta} \end{aligned}$$

where we have used the fact that $|\mathbb{E} [\widehat{p_m}] - p| \leq \tilde{c}_m h^\beta$ for some $\tilde{c}_m > 0$. \square

Next, we bound the variance of our estimator. This shows that the variance tends to zero as the number of samples grows large and the bandwidth shrinks.

Lemma 5.2. *The variance of the estimator $\widehat{p_1 \cdots p_M}(\theta)$ satisfies*

$$\sup_{p_1, \dots, p_M \in \mathcal{P}(\beta, L)} \mathbb{V} [\widehat{p_1 \cdots p_M}(\theta)] \leq \sum_{m=1}^M \binom{M}{m} \frac{c_m}{T^m h^{dm}}$$

for some $c_1, \dots, c_M > 0$ and $0 < h \leq 1$.

Proof. For all $p_1, \dots, p_M \in \mathcal{P}(\beta, L)$,

$$\begin{aligned} \mathbb{V} [\widehat{p_1 \cdots p_M}] &= \mathbb{E} [\widehat{p_1}^2] \cdots \mathbb{E} [\widehat{p_M}^2] - \mathbb{E} [\widehat{p_1}]^2 \cdots \mathbb{E} [\widehat{p_M}]^2 \\ &= \left(\prod_{m=1}^M \mathbb{V} [\widehat{p_m}] + \mathbb{E} [\widehat{p_m}]^2 \right) - \left(\prod_{m=1}^M \mathbb{E} [\widehat{p_m}]^2 \right) \\ &\leq \sum_{m=0}^{M-1} \binom{M}{m} \frac{\tilde{c}^m c^{M-m}}{T^{M-m} h^{d(M-m)}} \\ &\leq \sum_{m=1}^M \binom{M}{m} \frac{c_m}{T^m h^{dm}} \end{aligned}$$

where we have used the facts that $\mathbb{V}[\widehat{p}_m] \leq \frac{c}{Th^d}$ for some $c > 0$ and $\mathbb{E}[\widehat{p}_m]^2 \leq \tilde{c}$ for some $\tilde{c} > 0$. \square

Finally, we use the bias and variance bounds to bound the MSE, which shows that our estimator is consistent.

Theorem 5.3. *If $h \asymp T^{-1/(2\beta+d)}$, the mean-squared error of the estimator $\widehat{p_1 \cdots p_M}(\theta)$ satisfies*

$$\sup_{p_1, \dots, p_M \in \mathcal{P}(\beta, L)} \mathbb{E} \left[\int (\widehat{p_1 \cdots p_M}(\theta) - p_1 \cdots p_M(\theta))^2 d\theta \right] \leq \frac{c}{T^{2\beta/(2\beta+d)}}$$

for some $c > 0$ and $0 < h \leq 1$.

Proof. For all $p_1, \dots, p_M \in \mathcal{P}(\beta, L)$, using the fact that the mean-squared error is equal to the variance plus the bias squared, we have that

$$\begin{aligned} & \mathbb{E} \left[\int (\widehat{p_1 \cdots p_M}(\theta) - p_1 \cdots p_M(\theta))^2 d\theta \right] \\ & \leq \left(\sum_{m=1}^M c_m h^{m\beta} \right)^2 + \sum_{m=1}^M \binom{M}{m} \frac{\tilde{c}_m}{T^m h^{dm}} \\ & \leq k T^{-2\beta/(2\beta+d)} + \frac{\tilde{k}}{T^{1-d(2\beta+d)}} \quad (\text{for some } k, \tilde{k} > 0) \\ & \leq \frac{c}{T^{2\beta/(2\beta+d)}} \end{aligned}$$

for some $c_1, \dots, c_M > 0$ and $\tilde{c}_1, \dots, \tilde{c}_M > 0$. \square

6 Method Scope

The theoretical results and algorithms in this paper hold for posterior distributions over finite-dimensional real spaces. These include generalized linear models (e.g. linear, logistic, or Poisson regression), mixture models with known weights, hierarchical models, and (more generally) finite-dimensional graphical models with unconstrained variables. This also includes both unimodal and multimodal posterior densities (such as in Section 8.2). However, the methods and theory presented here do not yet extend to cases such as infinite dimensional models (e.g. nonparametric Bayesian models [5]) nor to distributions over the simplex (e.g. topics in latent Dirichlet allocation [3]). In the future, we hope to extend this work to these domains.

7 Related Work

In [19, 2, 16], the authors develop a way to sample approximately from a posterior distribution when only a small randomized mini-batch of data is used at each step. In [9], the authors used a hypothesis test to decide whether to accept or reject proposals using a small set of data (adaptively) as opposed to the exact

Metropolis-Hastings rule. This reduces the amount of time required to compute the acceptance ratio. Since all of these algorithms are still sequential, they can be directly used in our algorithm to generate subposterior samples to further speed up the entire sampling process.

Several parallel MCMC algorithms have been designed for specific models, such as for topic models [18, 14] and nonparametric mixture models [21]. These approaches still require synchronization to be correct (or approximately correct), while ours aims for more general model settings and does not need synchronization until the final combination stage.

Consensus Monte Carlo [17] is perhaps the most relevant work to ours. In this algorithm, data is also portioned into different machines and MCMC is performed independently on each machine. Thus, it roughly has the same time complexity as our algorithm. However, the prior is not explicitly reweighted during sampling as we do in Eq 1, and final samples for the full posterior are generated by averaging subposterior samples. Furthermore, this algorithm has few theoretical guarantees. We find that this algorithm can be viewed as a relaxation of our nonparametric, asymptotically exact sampling procedure, where samples are generated from an evenly weighted mixture (instead of each component having weight w_t .) and where each sample is set to θ_t instead of being drawn from $\mathcal{N}(\theta_t, \frac{h}{M} I_d)$. This algorithm is one of our experimental baselines.

8 Empirical Study

In the following sections, we demonstrate empirically that our method allows for quicker, MCMC-based estimation of a posterior distribution, and that our consistent-estimator-based procedures yield asymptotically exact results. We show our method on a few Bayesian models using both synthetic and real data. In each experiment, we compare the following strategies for parallel, communication-free sampling:⁴

- **Single chain full-data posterior samples (regularChain)**—Typical, single-chain MCMC for sampling from the full-data posterior.
- **Parametric subposterior density product estimate (parametric)**—For M sets of subposterior samples, the combination yielding samples from the parametric density product estimate.
- **Nonparametric subposterior density product estimate (nonparametric)**—For M sets of subposterior samples, the combination yielding samples from the nonparametric density product estimate.

⁴We did not directly compare with the algorithms that require synchronization since the setup of these experiments can be rather different. We plan to explore these comparisons in the extended version of this paper.

- **Semiparametric subposterior density product estimate** (`semiparametric`)—For M sets of subposterior samples, the combination yielding samples from the semiparametric density product estimate.
- **Subposterior sample average** (`subpostAvg`)—For M sets of subposterior samples, the average of M samples consisting of one sample taken from each subposterior.
- **Subposterior sample pooling** (`subpostPool`)—For M sets of subposterior samples, the union of all sets of samples.
- **Duplicate chains full-data posterior sample pooling** (`duplicateChainsPool`)—For M sets of samples from the full-data posterior, the union of all sets of samples.

To assess the performance of our sampling and combination strategies, we ran a single chain of MCMC on the full data for 500,000 iterations, removed the first half as burn-in, and considered the remaining samples the “groundtruth” samples for the true posterior density. We then needed a general method to compare the distance between two densities given samples from each, which holds for general densities (including multimodal densities, where it is ineffective to compare moments such as the mean and variance⁵). Following work in density-based regression [15], we use an estimate of the L_2 distance, $d_2(p, \hat{p})$, between the groundtruth posterior density p and a proposed posterior density \hat{p} , where $d_2(p, \hat{p}) = \|p - \hat{p}\|_2 = (\int (p(\theta) - \hat{p}(\theta))^2 d\theta)^{1/2}$.

In the following experiments involving timing, to compute the posterior L_2 error at each time point, we collected all samples generated before a given number of seconds, and added the time taken to transfer the samples and combine them using one of the proposed methods. In all experiments and methods, we followed a fixed rule of removing the first $\frac{1}{6}$ samples for burn-in (which, in the case of combination procedures, was applied to each set of subposterior samples before the combination was performed).

Experiments were conducted with a standard cluster system. We obtained subposterior samples by submitting batch jobs to each worker since these jobs are all independent. We then saved the results to the disk of each worker and transferred them to the same machine which performed the final combination.

8.1 Generalized Linear Models

Generalized linear models are widely used for many regression and classification problems. Here we conduct experiments, using logistic regression as a test case, on

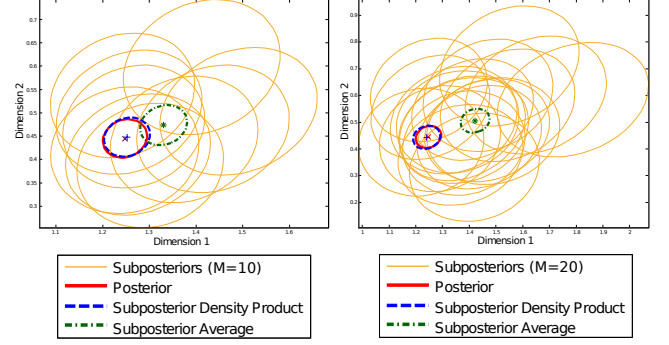


Figure 1: Bayesian logistic regression posterior ovals. We show the posterior 90% probability mass ovals for the first 2-dimensional marginal of the posterior, the M subposteriors, the subposterior density product (via the `parametric` procedure), and the subposterior average (via the `subpostAvg` procedure). We show $M=10$ subsets (left) and $M=20$ subsets (right). The subposterior density product generates samples that are consistent with the true posterior, while the `subpostAvg` produces biased results, which grow in error as M increases.

both synthetic and real data to demonstrate the speed of our parallel MCMC algorithm compared with typical MCMC strategies.

8.1.1 Synthetic data

Our synthetic dataset contains 50,000 observations in 50 dimensions. To generate the data, we drew each element of the model parameter β and data matrix X from a standard normal distribution, and then drew each outcome as $y_i \sim \text{Bernoulli}(\text{logit}^{-1}(X_i \beta))$ (where X_i denotes the i^{th} row of X)⁶. We use Stan, an automated Hamiltonian Monte Carlo (HMC) software package,⁷ to perform sampling for both the true posterior (for groundtruth and comparison methods) and for the subposteriors on each machine. One advantage of Stan is that it is implemented with C++ and uses the No-U-Turn sampler for HMC, which does not require any user-provided parameters [8].

In Figure 1, we illustrate results for logistic regression, showing the subposterior densities, the subposterior density product, the subposterior sample average, and the true posterior density, for the number of subsets M set to 10 (left) and 20 (right). Samples generated by our approach (where we draw samples from the subposterior density product via the `parametric` procedure) overlap with the true posterior much better than those generated via the `subpostAvg` (subposterior sample average) procedure—averaging of samples appears to create systematic biases. Further, the error in

⁵In these cases, dissimilar densities might have similar low-order moments. See Section 8.2 for an example.

⁶Note that we did not explicitly include the intercept term in our logistic regression model.

⁷<http://mc-stan.org>

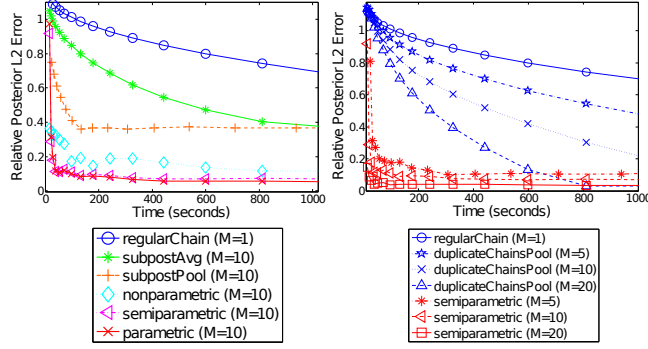


Figure 2: Posterior L_2 error vs time for logistic regression. Left: the three combination strategies proposed in this paper (**parametric**, **nonparametric**, and **semiparametric**) reduce the posterior error much more quickly than a single full-data Markov chain; the **subpostAvg** and **subpostPool** procedures yield biased results. Right: we compare with multiple full-data Markov chains (**duplicateChainsPool**); our method yields faster convergence to the posterior even though only a fraction of the data is being used by each chain.

averaging appears to increase as M grows. In Figure 2 (left) we show the posterior error vs time. A regular full-data chain takes much longer to converge to low error compared with our combination methods, and simple averaging and pooling of subposterior samples gives biased solutions.

We next compare our combination methods with multiple independent “duplicate” chains each run on the full dataset. Even though our methods only require a fraction of the data storage on each machine, we are still able to achieve a significant speed-up over the full-data chains. This is primarily because the duplicate chains cannot parallelize burn-in (i.e. each chain must still take some n steps before generating reasonable samples, and the time taken to reach these n steps does not decrease as more machines are added). However, in our method, each subposterior sampler can take each step more quickly, effectively allowing us to decrease the time needed for burn-in as we increase M . We show this empirically in Figure 2 (right), where we plot the posterior error vs time, and compare with full duplicate chains as M is increased.

Using a Matlab implementation of our combination algorithms, all (batch) combination procedures take under twenty seconds to complete on a 2.5GHz Intel Core i5 with 16GB memory.

8.1.2 Real-world data

Here, we use the *covtype* (predicting forest cover types)⁸ dataset, containing 581,012 observations in 54 dimen-

⁸<http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets>

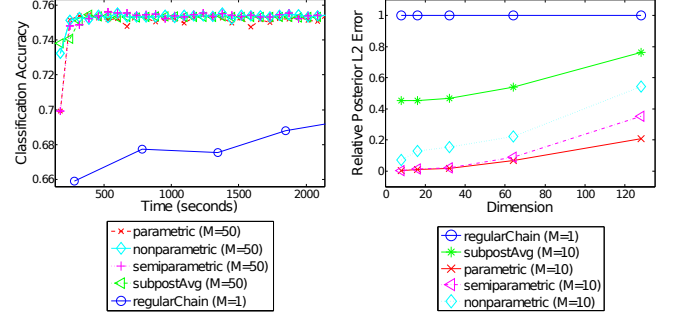


Figure 3: Left: Bayesian logistic regression classification accuracy vs time for the task of predicting forest cover type. Right: Posterior error vs dimension on synthetic data at 1000 seconds, normalized so that **regularChain** error is fixed at 1.

sions. A single chain of HMC running on this entire dataset takes an average of 15.76 minutes per sample; hence, it is infeasible to generate groundtruth samples for this dataset. Instead we show classification accuracy vs time. For a given set of samples, we perform classification using a sample estimate of the posterior predictive distribution for a new label y with associated datapoint x , i.e.

$$P(y|x, y^N, x^N) = \int P(y|x, \beta, y^N, x^N) P(\beta|x^N, y^N) \\ \approx \frac{1}{S} \sum_{s=1}^S P(y|x, \beta_s)$$

where x^N and y^N denote the N observations, and $P(y|x, \beta_s) = \text{Bernoulli}(\text{logit}^{-1}(x^\top \beta_s))$. Figure 3 (left) shows the results for this task, where we use $M=50$ splits. The parallel methods achieve a higher accuracy much faster than the single-chain MCMC algorithm.

8.1.3 Scalability with dimension

We investigate how the errors of our methods scale with dimensionality, to compare the different estimators implicit in the combination procedures. In Figure 3 (right) we show the relative posterior error (taken at 1000 seconds) vs dimension, for the synthetic data with $M=10$ splits. The errors at each dimension are normalized so that the **regularChain** error is equal to 1. Here, the **parametric** (asymptotically biased) procedure scales best with dimension, and the **semiparametric** (asymptotically exact) procedure is a close second. These results also demonstrate that, although the **nonparametric** method can be viewed as implicitly sampling from a nonparametric density estimate (which is usually restricted to low-dimensional densities), the performance of our method does not suffer greatly when we perform parallel MCMC on posterior distributions in much higher-dimensional spaces.

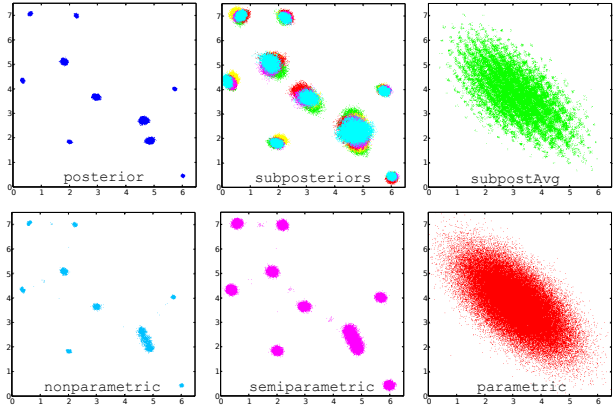


Figure 4: Gaussian mixture model posterior samples. We show 100,000 samples from a single 2-d marginal (corresponding to the posterior over a single mean parameter) of the full-data posterior (top left), all subposteriors (top middle—each one is given a unique color), the subposterior average via the **subpostAvg** procedure (top right), and the subposterior density product via the **nonparametric** procedure (bottom left), **semiparametric** procedure (bottom middle), and **parametric** procedure (bottom right).

8.2 Gaussian mixture models

In this experiment, we aim to show correct posterior sampling in cases where the full-data posterior, as well as the subposteriors, are multimodal. We will see that the combination procedures that are asymptotically biased suffer greatly in these scenarios. To demonstrate this, we perform sampling in a Gaussian mixture model. We generate 50,000 samples from a ten component mixture of 2-d Gaussians. The resulting posterior is multimodal; this can be seen by the fact that the component labels can be arbitrarily permuted and achieve the same posterior value. For example, we find after sampling that the posterior distribution over each component mean has ten modes. To sample from this multimodal posterior, we used the Metropolis-Hastings algorithm, where the component labels were permuted before each step (note that this permutation results in a move between two points in the posterior distribution with equal probability).

In Figure 4 we show results for $M=10$ splits, showing samples from the true posterior, overlaid samples from all five subposteriors, results from averaging the subposterior samples, and the results after applying our three subposterior combination procedures. This figure shows the 2-d marginal of the posterior corresponding to the posterior over a single mean component. The **subpostAvg** and **parametric** procedures both give biased results, and cannot capture the multimodality of the posterior. We show the posterior error vs time in Figure 5 (left), and see that our asymptotically exact

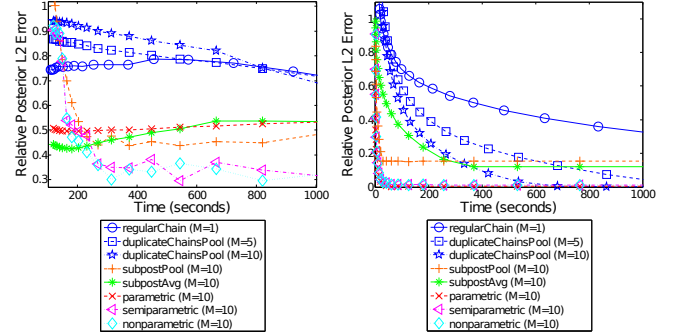


Figure 5: Left: Gaussian mixture model posterior error vs time results. Right: Poisson-gamma hierarchical model posterior error vs time results.

methods yield quick convergence to low posterior error.

8.3 Hierarchical models

We show results on a hierarchical Poisson-gamma model of the following form

$$\begin{aligned} a &\sim \text{Exponential}(\lambda) & b &\sim \text{Gamma}(\alpha, \beta) \\ q_i &\sim \text{Gamma}(a, b) & x_i &\sim \text{Poisson}(q_i t_i) \quad i = 1, \dots, N \end{aligned}$$

for $N=50,000$ observations. We draw $\{x_i\}_{i=1}^N$ from the above generative process (after fixing values for a , b , λ , and $\{t_i\}_{i=1}^N$), and use $M=10$ splits. We again perform MCMC using the Stan software package.

In Figure 5 (right) we show the posterior error vs time, and see that our combination methods complete burn-in and converge to a low posterior error very quickly relative to the **subpostAvg** and **subpostPool** procedures and full-data chains.

9 Discussion and Future Work

In this paper, we present an embarrassingly parallel MCMC algorithm and provide theoretical guarantees about the samples it yields. Experimental results demonstrate our method’s potential to speed up burn-in and perform faster asymptotically correct sampling. Further, it can be used in settings where data are partitioned onto multiple machines that have little intercommunication—this is ideal for use in a MapReduce setting. Currently, our algorithm works primarily when the posterior samples are real, unconstrained values and we plan to extend our algorithm to more general settings in future work.

10 Acknowledgments

This work is supported in part by DARPA X Data FA87501220324, NIH GWAS R01GM087694, and NSF Social Media IIS1111142.

References

- [1] Alekh Agarwal and John C Duchi, *Distributed delayed stochastic optimization*, Decision and Control (CDC), 2012 IEEE 51st Annual Conference on, IEEE, 2012, pp. 5451–5452.
- [2] Sungjin Ahn, Anoop Korattikara, and Max Welling, *Bayesian posterior sampling via stochastic gradient fisher scoring*, Proceedings of the 29th International Conference on Machine Learning, 2012, pp. 1591–1598.
- [3] David M Blei, Andrew Y Ng, and Michael I Jordan, *Latent dirichlet allocation*, The Journal of Machine Learning Research **3** (2003), 993–1022.
- [4] Jeffrey Dean and Sanjay Ghemawat, *Mapreduce: simplified data processing on large clusters*, Communications of the ACM **51** (2008), no. 1, 107–113.
- [5] Samuel J Gershman and David M Blei, *A tutorial on bayesian nonparametric models*, Journal of Mathematical Psychology **56** (2012), no. 1, 1–12.
- [6] Nils Lid Hjort and Ingrid K Glad, *Nonparametric density estimation with a parametric start*, The Annals of Statistics (1995), 882–904.
- [7] Qirong Ho, James Cipar, Henggang Cui, Seunghak Lee, Jin Kyu Kim, Phillip B. Gibbons, Gregory R. Ganger, Garth Gibson, and Eric P. Xing, *More effective distributed ml via a stale synchronous parallel parameter server*, Advances in Neural Information Processing Systems, 2013.
- [8] Matthew D Hoffman and Andrew Gelman, *The no-urn sampler: Adaptively setting path lengths in hamiltonian monte carlo*, arXiv preprint arXiv:1111.4246 (2011).
- [9] Anoop Korattikara, Yutian Chen, and Max Welling, *Austerity in MCMC land: Cutting the Metropolis-Hastings budget*, arXiv preprint arXiv:1304.5299 (2013).
- [10] John Langford, Alex J Smola, and Martin Zinkevich, *Slow learners are fast*, Advances in Neural Information Processing Systems, 2009.
- [11] Kathryn Blackmond Laskey and James W Myers, *Population Markov chain Monte Carlo*, Machine Learning **50** (2003), no. 1-2, 175–196.
- [12] Lucien Le Cam, *Asymptotic methods in statistical decision theory*, New York (1986).
- [13] Lawrence Murray, *Distributed Markov chain Monte Carlo*, Proceedings of Neural Information Processing Systems Workshop on Learning on Cores, Clusters and Clouds, vol. 11, 2010.
- [14] David Newman, Arthur Asuncion, Padhraic Smyth, and Max Welling, *Distributed algorithms for topic models*, The Journal of Machine Learning Research **10** (2009), 1801–1828.
- [15] Junier Oliva, Barnabás Póczos, and Jeff Schneider, *Distribution to distribution regression*, Proceedings of The 30th International Conference on Machine Learning, 2013, pp. 1049–1057.
- [16] Sam Patterson and Yee Whye Teh, *Stochastic gradient riemannian langevin dynamics on the probability simplex*, Advances in Neural Information Processing Systems, 2013.
- [17] Steven L. Scott, Alexander W. Blocker, and Fernando V. Bonassi, *Bayes and big data: The consensus monte carlo algorithm*, Bayes 250, 2013.
- [18] Alexander Smola and Shravan Narayanamurthy, *An architecture for parallel topic models*, Proceedings of the VLDB Endowment **3** (2010), no. 1-2, 703–710.
- [19] Max Welling and Yee W Teh, *Bayesian learning via stochastic gradient Langevin dynamics*, Proceedings of the 28th International Conference on Machine Learning, 2011, pp. 681–688.
- [20] Darren J Wilkinson, *Parallel Bayesian computation*, Statistics Textbooks and Monographs **184** (2006), 477.
- [21] Sinead Williamson, Avinava Dubey, and Eric P Xing, *Parallel Markov chain Monte Carlo for nonparametric mixture models*, Proceedings of the 30th International Conference on Machine Learning, 2013, pp. 98–106.

Modeling Citation Networks Using Latent Random Offsets

Willie Neiswanger[§]
willie@cs.cmu.edu

Chong Wang[†]
chongw@cs.princeton.edu*

Qirong Ho[§]
qho@cs.cmu.edu

Eric P. Xing[§]
epxing@cs.cmu.edu

[§]Machine Learning Department, Carnegie Mellon University

[†]Voleon Capital Management

Abstract

Out of the many potential factors that determine which links form in a document citation network, two in particular are of high importance: first, a document may be cited based on its subject matter—this can be modeled by analyzing document content; second, a document may be cited based on which other documents have previously cited it—this can be modeled by analyzing citation structure. Both factors are important for users to make informed decisions and choose appropriate citations as the network grows. In this paper, we present a novel model that integrates the merits of content and citation analyses into a single probabilistic framework. We demonstrate our model on three real-world citation networks. Compared with existing baselines, our model can be used to effectively explore a citation network and provide meaningful explanations for links while still maintaining competitive citation prediction performance.

1 Introduction

Many large citation networks—Wikipedia, arXiv, and PubMed¹, to name a few—continue to quickly grow in size, and the structure of these networks continues to increase in complexity. To effectively explore large-scale and complex data like these and extract useful information, users rely more and more on various types of guidance for help. An important type of guidance comes from the citations (or links) in the network. Citations serve as paths that users can easily follow, and do not require users to specify certain keywords in advance. In scientific research, for example, researchers

often find potentially interesting articles by following citations made in other articles. In Wikipedia, users often find explanations of certain terms by following the links made by other Wikipedia users. Thus, generating relevant citations is important for many users who may frequently rely on these networks to explore data and find useful information.

We believe that, among many, two important factors largely determine how a document citation network is formed: the documents’ contents and the existing citation structure. Take as an example a citation network of computer science articles. A research paper about “support vector machines (SVMs)”, for instance, might be cited by several other articles that develop related methods, based on the subject matter alone. This type of information can be well captured by analyzing the content of the documents. However, the existing citation structure is also important. If this SVM paper included great results on a computer vision dataset, for example, it might be cited by many vision papers that are *not* particularly similar in content. Though different in content, this SVM paper could be very important to users in a different topic area, and should be considered by these users when choosing citations. This type of information cannot be easily captured by analyzing document content, but can be discovered by analyzing the existing citation structure among documents while studying the contents of the papers that generated these citations.

Given these observations, we present a probabilistic model to accurately model citation networks by integrating content and citation/link information into a single framework. We name our approach a *latent random offset* (LRO) model. The basic idea is as follows: we first represent the content of each document using a latent vector representation (i.e. “topics”) that summarizes the document content. Then, each latent representation is augmented in an additive manner with a random offset vector; this vector models information from the citation structure that is not well captured

* Work completed while at Carnegie Mellon University.

¹<http://www.wikipedia.org/>, <http://arxiv.org/>, and <http://www.ncbi.nlm.nih.gov/pubmed>

Sistine Chapel (Simple English Wikipedia)

Text: "The Sistine Chapel is a large chapel in the Vatican Palace, the place in Italy where the Pope lives. The Chapel was built between 1473 and 1481 by Giovanni dei Dolci for Pope Sistus IV...The Sistine Chapel is famous for its fresco paintings by the Renaissance painter Michelangelo..."

In-Links (Citing Documents): (1) Raphael, (2) Ten Commandments, (3) Chapel, (4) Apostolic Palace, (5) St. Peter's Basilica

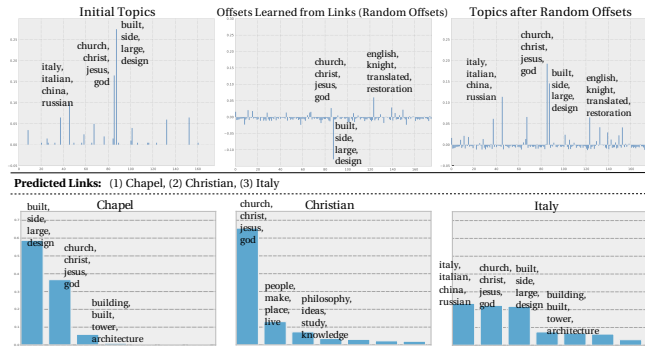


Figure 1: Analysis of content, latent offsets, and predicted links for the *Sistine Chapel* document in the Simple English Wikipedia dataset. The first row shows an example passage from the document. The next row shows the names of the documents that cite *Sistine Chapel*. The next row shows the initial latent topics (first column), the latent offsets learned from links (second column), and the latent topics after applying the offsets (third column). The final row shows interpretable link predictions; for each predicted link, we show the relative weight that each latent topic contributed to the prediction.

by document content. The final augmented representation is then used to model how this document is cited by other documents. To motivate this representation, we present sample outputs from running LRO on the Simple English Wikipedia.

Examples from Simple English Wikipedia.

The first graph in the top row of Figure 1 shows, for the *Sistine Chapel* article in the Simple English Wikipedia, the latent vector representation, which is concentrated around three topics: **countries** (*italy, italian, china, russian*), **Christianity** (*church, christ, jesus, god*), and **architecture** (*built, side, large, design*). Here we've listed the top four words in each topic (in parens). The incoming links to the *Sistine Chapel* article are also shown; these citing documents determine the random offsets for *Sistine Chapel*. The random offsets can be thought of as "corrections" to the latent vector representation, based on the content of citing documents—for example, the two largest positive offsets are **Christianity** (*church, christ, jesus, god*) and **Anglicanism** (*english, knight, translated, restoration*), meaning that the citing documents strongly exhibit these two topics (compared to the *Sistine Chapel* article). On the other hand, there is a large negative offset on **architecture** (*built, side, large, design*), indicating that the citing documents do not exhibit this topic as

much as *Sistine Chapel*.

Notably, the topic **Anglicanism** (containing words related to Christianity in England) is found in the random offsets for *Sistine Chapel*, but is absent from its latent vector representation. This is because the Sistine Chapel is in the Vatican City, and thus its article does not emphasize content relating to England or Anglicanism (even though they are all related to Christianity). However, documents that link to *Sistine Chapel*, such as *Chapel*, talk about the Anglican Church in England. This is an example where pertinent information is found in the citation structure, but not in the document content. By capturing this citation information, the LRO model provides insights into the context surrounding a document.

Following this idea, we can add the latent vector and random offsets together to obtain the "augmented representation" of a document (i.e. the "topics after random offsets" graph in Figure 1), which takes into account not just its content, but the content of its citing documents as well. Link predictions in the LRO model are based upon the intuition that a document i cites document j only if both documents have similar representations. This intuition is captured in the bottom row of graphs in Figure 1, which explains three out-links predicted by the LRO model for the *Sistine Chapel* document. For each predicted link, we show the topics that contributed most to the prediction, and not surprisingly, the most important topics for each link also feature strongly in the augmented representation for the *Sistine Chapel*. Knowing which topics contributed to the prediction of links not only helps users interpret existing links within a document corpus, but also gives users an explanation for every new link predicted by the LRO model—for instance, a user might invoke LRO to recommend citations for an academic paper, and such "link explanations" give the user a quick overview of why each recommendation is relevant.

We note that of the three predicted out-links for *Sistine Chapel*, two of them (*Chapel, Italy*) are actual out-links in *Sistine Chapel*, while the third, *Christian*, is obviously relevant but not found in the document. This motivates another application of LRO: predicting relevant but missing links in document corpora; in this case, we are completing the references for a Wikipedia article. Another application context is academic paper writing: LRO can be used to recommend important (but otherwise overlooked) citations for a newly-written academic paper.

The rest of this paper is organized as follows: we begin by formalizing latent random offset modeling, and then show how we can use it to model citation networks. We then develop a fast learning algorithm

with linear complexity in the size of the number of citations, and empirically evaluate our approach using three real-world citation networks. Compared with several baselines, our model not only improves citation prediction performance, but also provides meaningful explanations for citations within the networks. By studying latent random offset representations, we show these explanations can be used to effectively interpret why our model predicts links for given documents and to explore citation networks.

2 Latent Random Offset Models

We introduce the general framework of latent random offsets for citation network modeling. Suppose our citation network consists of D documents (i.e. nodes), $\mathcal{D} = \{x_1, x_2, \dots, x_D\}$. We use $y_{ij} = 1$ or 0 to indicate whether document i cites document j or not. Note that y_{ij} is *directed*, meaning y_{ij} is not necessarily the same as y_{ji} .

Each document x_j is usually a high-dimensional vector in \mathbb{R}^V , where V is the vocabulary size, so it is desirable to represent x_j using a low-dimensional vector θ_j . In other words, the mapping

$$\theta_j = \theta_j(x_j) \quad (1)$$

serves as a summarization of the original document content x_j , and these summarizations can be used to measure the content similarities of different documents.

However, in real citation networks, a document can be cited by others for reasons outside of its content information. For example, a target document might provide an influential idea that can be used in many different fields and thus be cited by a diverse set of documents. This information is encoded not in the document content but in the citation network structure. We choose to model this phenomenon by allowing a random offset vector ϵ_j to augment the low-dimensional vector θ_j , which gives the augmented representation

$$v_j = \theta_j + \epsilon_j. \quad (2)$$

The offset vector ϵ_j is used to capture the network structure information that is *not* contained in the document's content. One important property of this augmented representation is that the random offset ϵ_j is aligned in the same space as θ_j . If the dimension of θ_j has some semantic explanations, then ϵ_j can be understood as modifications of those explanations.

Finally we consider using a function f to model the citation from document i to document j , such that

$$f(\theta_i, \theta_j + \epsilon_j) \approx y_{ij} \quad (\text{for all } i, j)$$

where y_{ij} is the citation indicator from document i to document j . Notice the asymmetric structure here for document i and j —we do not consider the offset vector ϵ_i for document i in our function f . In real citation networks, when a new document joins the citation network by citing some other documents, this new document is effectively “not in” the network. It will be most likely to cite other documents based only on their content and their citations, as no network information exists for this new document. One advantage of this formulation is that we can make citation predictions for a brand new document by only using its content information.

In the next two sections, we first describe how we create the low-dimensional document content representation θ_j and how we use the latent random offset model for citation network modeling.

2.1 Probabilistic topic models for document content representation

There are many potential ways to create the low-dimensional document content representation described in Eq. 1. Here we choose to use probabilistic topic models. Topic models [5] are used to discover a set of “topics” (or themes) from a large collection of documents. These topics are distributions over terms, which are biased to be associated under a single theme. One notable property of these models is that they often provide an interpretable low-dimensional representation of the documents [10]. They have been used for tasks like corpus exploration [8], information retrieval [23] and recommendation [22].

Here we describe the simplest topic model, latent Dirichlet allocation (LDA) [7] and use it to create the low-dimensional document content representations. Assume there are K topics, β_k , $k = 1, \dots, K$ and each β_k is a distribution over a fixed vocabulary. For each document j , the generative process is as follows,

1. Draw topic proportions $\theta_j \sim \text{Dirichlet}(\alpha)$
2. For each word x_{jn} in document j ,
 - (a) Draw topic assignment $z_{jn} \sim \text{Mult}(\theta_j)$
 - (b) Draw word $x_{jn} \sim \text{Mult}(\beta_{z_{jn}})$

This process describes how the words of a document are generated from a mixture of topics that are shared by the corpus. The topic proportions θ_j are document-specific and we use these topic proportions as our low-dimensional document content representation.

Given a document collection, the only observations are the words in the documents. The topics, topic proportions for each document, and topic assignments for each word, are all latent variables that have to be

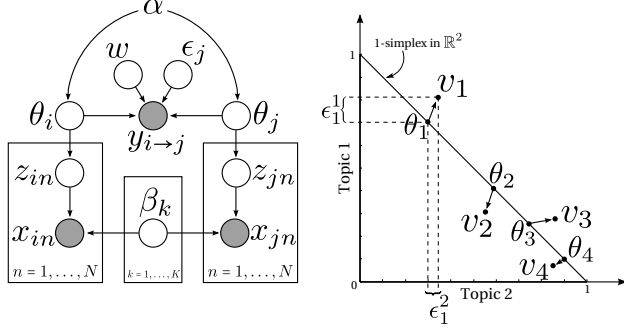


Figure 2: Left: The LRO graphical model. Only two documents (i and j) and one citation (from i to j) are shown. The augmented latent representation representation for document j is $v_j = \theta_j + \epsilon_j$. Right: An illustration of the random offsets. We show each document’s content vector θ_j (which lies on the simplex), its offsets ϵ_j due to link structure (the superscript indicates the dimension for ϵ_j), and the resulting augmented latent representation v_j .

determined from the data. LDA has been extensively studied in the literature and many efficient algorithms have been proposed to fit the LDA model variables [7, 12, 21]. For example, standard learning algorithms like variational EM or Gibbs sampling can be used to estimate these quantities [7]. These methods give us the estimated document content representations θ_j in terms of an approximate posterior distribution or point estimates.

2.2 Modeling citations via random offsets

Having described how we represent the documents in a low dimensional space, we now consider how to create the augmented representations introduced in Eq. 2. We model our latent random offset vector ϵ_j with a multivariate Gaussian distribution

$$\epsilon_j \sim \mathcal{N}(0, \lambda^{-1} I_K).$$

where λ is a scalar precision parameter for the latent random offsets.

Using the general idea of latent random offset modeling shown in Eq. 2 and probabilistic topic models described in Section 2.1, our *latent random offset model* (LRO) for citation network modeling has the following generative process (Figure 2 shows the graphical model). Assuming K topics, $\beta_{1:K}$,

1. For each document j ,
 - (a) Draw topic proportions $\theta_j \sim \text{Dirichlet}(\alpha)$
 - (b) Draw latent random offset $\epsilon_j \sim \mathcal{N}(0, \lambda^{-1} I_K)$ and set the document augmented representation as $v_j = \theta_j + \epsilon_j$

- (c) For each word x_{jn} ,
 - i. Draw topic assignment $z_{jn} \sim \text{Mult}(\theta)$
 - ii. Draw word $x_{jn} \sim \text{Mult}(\beta_{z_{jn}})$

2. For each *directed* pair of documents (i, j) , draw the citation indicator

$$y_{ij} \sim \mathcal{N}(y | w\theta_i^\top v_j, \tau_{ij}^{-1}).$$

where $w \in \mathbb{R}_+$ is a global scaling parameter to account for potential inefficiencies of the topic proportions θ_i , which are constrained to the simplex.² We chose a Gaussian response to model the citations, in similar fashion to [22]. Notation τ_{ij}^{-1} is the precision parameter for the Gaussian distribution. Here, we choose to stray from a formal generative process and also treat the y_{ij} as parameters, such that τ_{ij} satisfies

$$\tau_{ij} = \begin{cases} \tau_1 & \text{if } y_{ij} = 1 \\ \tau_0 & \text{if } y_{ij} = 0. \end{cases}$$

In this formulation, τ_1 specifies the precision if a link exists from document i to j , while τ_0 is for the case where the link does not exist. We set τ_0 to be much smaller (i.e. higher noise) than τ_1 — this is similar to the assumption made in [22], which models the fact that $y_{ij} = 0$ could either mean it is not appropriate for document i to cite document j , or simply that document i should cite document j but has inadvertently neglected to cite it. This also enables a fast learning algorithm with complexity linear in the number of citations (See Section 3 for details).

The expectation of the citation can be computed as

$$\mathbb{E}[y_{ij}] = w\theta_i^\top v_j = w(\theta_i^\top \theta_j) + w(\theta_i^\top \epsilon_j).$$

This reveals how likely it is for a citation from document i to document j to occur under our model. If the documents have similar content or document j has certain large positive offsets, it is more likely to be cited by document i .

For a document j , our latent representation θ_j is over a simplex. In Figure 2 (right), we show how the random offsets ϵ_j produce the augmented representation v_j .

2.3 Citation prediction

In a system for citation prediction, it is more realistic to suggest citations than to make hard decisions for the users. This is common in many recommender systems [13, 22]. For a particular document i , we rank the potential citations according to the score

$$S_{ij} = w\theta_i^\top v_j,$$

²Our experiments show that optimizing the global scaling parameter w is important for obtaining good results.

for all other documents j , and suggest citations based on this score (excluding document i and all pre-existing citations).

3 Learning Algorithm

We use maximum a posteriori (MAP) estimation to learn the latent parameters of the LRO, where we perform a coordinate ascent procedure to carry out the optimization. Maximization of the posterior is equivalent to maximizing the complete log likelihood of $v_{1:D}$, $\theta_{1:D}$ and $\beta_{1:K}$, which we can write as

$$\mathcal{L} = -\frac{\lambda}{2} \sum_j (v_j - \theta_j)^\top (v_j - \theta_j) - \sum_{i \neq j} \frac{\tau_{ij}}{2} (y_{ij} - w \theta_i^\top v_j)^2 + \sum_j \sum_n \log \left(\sum_k \theta_{jk} \beta_{k, x_{jn}} \right).$$

where we have omitted a constant and set $\alpha = 1$.

First, given topics $\beta_{1:K}$ and augmented representations $v_{1:D}$, for all documents, we describe how to learn the topic proportions θ_j . We first define $\phi_{jnk} = q(z_{jn} = k)$. Then we separate the items that contain θ_j and apply Jensen's inequality,

$$\begin{aligned} \mathcal{L}(\theta_j) &\geq -\frac{\lambda}{2} \sum_j (v_j - \theta_j)^\top (v_j - \theta_j) \\ &\quad + \sum_n \sum_k \phi_{jnk} (\log \theta_{jk} \beta_{k, x_{jn}} - \log \phi_{jnk}) \\ &= \mathcal{L}(\theta_j, \phi_j). \end{aligned}$$

where $\phi_j = (\phi_{jnk})_{n=1, k=1}^{D \times K}$. The optimal ϕ_{jnk} then satisfies

$$\phi_{jnk} \propto \theta_{jk} \beta_{k, x_{jn}}.$$

The $\mathcal{L}(\theta_j, \phi_j)$ gives the *tight* lower bound of $\mathcal{L}(\theta_j)$. We cannot optimize θ_j analytically, but we can use the projection gradient [3] method for optimization.³

Second, given this ϕ , we can optimize the topics $\beta_{1:K}$ with

$$\beta_{kx} \propto \sum_j \sum_n \phi_{jnk} 1[x_{jn} = x].$$

This is the same M-step update for topics as in LDA [7].

Next, we would like to optimize the augmented representations $v_{1:D}$. We can write the component of the log likelihood with terms containing v_j as

$$\begin{aligned} \mathcal{L}(v_j) &= -\frac{\lambda}{2} (v_j - \theta_j)^\top (v_j - \theta_j) \\ &\quad - \sum_{i, i \neq j} \frac{\tau_{ij}}{2} (y_{ij} - w \theta_i^\top v_j)^2. \end{aligned}$$

³On our data, we found that simply fixing θ_j as the estimate from the LDA model gives comparable performance and saves computation.

To maximize this quantity, we take the gradient of $\mathcal{L}(v_j)$ with respect to v_j and set it to 0, which gives an update for v_j

$$\begin{aligned} v_j^* &\leftarrow \left(\lambda I_K + w^2 \left((\tau_1 - \tau_0) \sum_{i \in \{i: i \rightarrow j\}} \theta_i \theta_j^\top + \tau_0 \sum_{i, i \neq j} \theta_i \theta_j^\top \right) \right)^{-1} \\ &\quad \times \left(\theta_j + w \tau_1 \sum_{i \in \{i: i \rightarrow j\}} \theta_i \right) \end{aligned} \quad (3)$$

where $\{i : i \rightarrow j\}$ denotes the set of documents that cite document j . For the second line of Eq. 3, we can see that the augmented representation v_j is affected by two main parts: the first is the content from document j (topic proportions θ_j) and the second is the content from other documents who cite document j (topic proportions θ_i , where $i \in \{i : i \rightarrow j\}$).

Next, we want to optimize the global scaling variable w . Isolating the terms in the complete log likelihood that contain w gives

$$\mathcal{L}(w) = - \sum_{i \neq j} \frac{\tau_{ij}}{2} (y_{ij} - w \theta_i^\top v_j)^2.$$

In a similar manner as the previous step, to maximize this quantity we take the gradient of $\mathcal{L}(w)$ with respect to w and set it to 0, which gives its update⁴

$$\begin{aligned} w^* &\leftarrow \left(\sum_j \left((\tau_1 - \tau_0) \sum_{i \in \{i: i \rightarrow j\}} (\theta_i^\top v_j)^2 + \tau_0 \sum_{i, i \neq j} (\theta_i^\top v_j)^2 \right) \right)^{-1} \\ &\quad \times \left(\tau_1 \sum_j \sum_{i \in \{i: i \rightarrow j\}} \theta_i^\top v_j \right). \end{aligned} \quad (4)$$

Empirically, we found that an optimal trade-off between computation time and performance involves performing LDA [7] initially to learn the latent representations θ_j , and then performing coordinate ascent to learn the augmented representations v_j and global parameter w . We detail this procedure in Algorithm 1.

Computational efficiency. We now show that our learning algorithm (Algorithm 1) has runtime complexity linear in the number of documents and citations.

First, estimating the topic proportions θ_j , $j = 1, \dots, D$ has the same complexity as the standard learning algorithm for LDA, which is linear in the number of documents.

Second, the augmented representations v_j , $j = 1, \dots, D$ and global scaling parameter w can be estimated in linear time, via a caching strategy — this is similar to the method adopted by [13, 22]. We now describe this strategy.

⁴In theory, this update could lead to a negative value. However, in our experiments, we did not see this happen.

Algorithm 1 MAP Parameter Learning

Input: A citation network of documents $\{x_j\}_{j=1}^D$ with directed links y_{ij} for $i, j \in \{1, \dots, D\}$, and stopping criteria δ

Output: Latent content representations θ_j , link-offset representations v_j , and global scale parameter w

- 1: Run LDA [7] on $\{x_j\}_{j=1}^D$ to learn $\theta_{1:D}$
 - 2: Initialize $v_{1:D} = \theta_{1:D}$ and $\text{eps} = \infty$
 - 3: **while** $\text{eps} > \delta$ **do**
 - 4: Update $w \leftarrow w^*$ ▷ Equation 4
 - 5: **for** $j = 1$ **to** D **do**
 - 6: Update $v_j \leftarrow v_j^*$ ▷ Equation 3
 - 7: **end for**
 - 8: Set $\text{eps} \leftarrow \|v_{1:D} - \tilde{v}_{1:D}\|$
 - 9: **end while**
-

For the augmented representation v_j (Eq. 3), we cache $\theta_0 = \sum_i \theta_i$. This allows us to update v_j (Eq. 3) using the identity

$$\sum_{i, i \neq j} \theta_i = \theta_0 - \theta_j.$$

Every time we update a θ_j , we also update the cache θ_0 , and this takes constant time w.r.t. the number of documents and citations.

For the global scaling parameter w (Eq. 4), we can compute

$$\begin{aligned} \sum_{i, i \neq j} (\theta_i^\top v_j)^2 &= \sum_{i, i \neq j} v_j^\top \theta_i \theta_i^\top v_j \\ &= v_j^\top (\sum_{i, i \neq j} \theta_i \theta_i^\top) v_j \\ &= v_j^\top (\sum_i \theta_i \theta_i^\top) v_j - v_j^\top \theta_j \theta_j^\top v_j \end{aligned}$$

in $O(K^2)$ time (constant in the number of docs and citations) by simply caching $\Theta_0 = \sum_i \theta_i \theta_i^\top$. This cache variable also requires $O(K^2)$ time to update whenever we modify some θ_j .

The remaining sums in Eqs 3,4 touch every citation exactly once, therefore a single update sweep over all v_j and w only requires constant work per edge (treating K as constant). We have therefore shown that Algorithm 1 is linear in the number of documents and citations. Moreover, we have attained linear scalability without resorting to treating missing citations as hidden data. This gives our LRO a data advantage over methods that hide missing citations, such as the RTM [9].

4 Related Work

Our proposed work focuses on two aspects of citation network modeling: 1) network understanding/exploration and 2) citation prediction. We therefore divide the related work section into these two categories.

Network understanding/exploration.

Network exploration is a broad empirical task concerned with, amongst other things, understanding the overall structure of the network [19], understanding the context of individual nodes [2], and discovering anomalous nodes or edges [20]. In addition to methods that operate on purely graph data, there are techniques that leverage both the graph as well as textual content, such as relational topic models (RTM) [9], Link-PLSA-LDA [17], and TopicFlow [18]. The idea behind such hybrid methods is that text and graph data are often orthogonal, providing complementary insights [11].

Our LRO model incorporates network information by modeling per-document random offsets that capture topical information from connected neighbors. These random offsets represent relevant topics that would otherwise not be found in the documents through content analysis. The Simple English Wikipedia analysis from the introduction provides a good example: the *Sistine Chapel* article’s random offsets (the top row of Figure 1) contain the topic **Anglicanism** (which is also related to Christianity), even though the article text’s latent topic representation makes no mention of it. In this manner, the LRO model helps us understand the context of network nodes (a.k.a. documents), and helps us to detect anomalous nodes (such as documents whose random offsets diverge greatly from their latent topic vectors).

Citation prediction. The citation prediction task can be approached by considering text features, network features, or a combination of both. In the text-only setting, approaches based on common text features (e.g., TF-IDF scores [4]) and latent space models (e.g., topic models [5]) can be used to measure similarities between two documents, allowing for ranking and prediction. However, text-only approaches cannot account for citation behavior due to the network structure.

In the network-only setting without document content, there are a number of commonly-used measures of node similarity, such as the Jaccard Coefficient, the Katz measure [14] and the Adamic/Adar measure [1]. Latent space models such as matrix factorization (MF) methods [15] can be used here. However, when test documents are out-of-sample with respect to the network (when we consider newly-written papers with no preexisting citations), these measures are inapplicable.

Finally, there are methods that combine both document content and network structure to predict citations. One such method is the relational topic models (RTM) [9], in which link outcomes depend on a reweighted inner product between latent positions (under the LDA model). The weights are learned for each latent dimension (topic), but are not specific to any document,

and thus only capture network behavior due to topic-level interactions. In contrast, our random offsets are learned on a per-document basis, capturing interaction patterns specific to each document, which in turn yields better predictive performance as shown in our empirical study. In [16], in addition to the document content, author information is also considered to model the citation structure. In [17], citations were treated as a parallel document (of citations) as to the document content of words. Neither of these methods use per-document offsets to model citation structure.

5 Empirical Study

We will empirically demonstrate the use of our model for modeling citation networks. We will first show quantitative results for citation prediction then present qualitative results using our model to explore citation networks.

Datasets. We use three citation network datasets,

1. The ACL Anthology paper citation network (*ACL*) contains 16,589 documents and 94,973 citations over multiple decades.
2. The arXiv high energy physics citation network (*arXiv*) contains 34,546 arXiv/hep-th articles and 421,578 citations from January 1993 through April 2003.
3. The Simple English Wikipedia citation network (*Wikipedia*) contains 27,443 articles, and 238,957 citations corresponding to user-curated hyperlinks between articles.

5.1 Citation prediction

For citation prediction, we compare against the RTM [9], matrix factorization (MF) [15], LDA-based predictions [7], and three common baseline algorithms. A detailed description is given below.

The first task is predicting held-out citations. Here we used a five-fold cross validation: for each document that has cited more than 5 documents, we held out 20% of the documents into test set and the rest into the training set.

The second task is predicting citations for new documents. To simulate this scenario, we train our model using all the citations before a certain year and predict the citations of the new documents published in that year. This task is important for a real citation prediction system, where user may input some text without existing citations. For this experiment, we excluded MF from the comparisons, because it cannot perform this task.

Evaluation metric. Our goal is to make citation predictions, where it is more realistic to provide a rank list of citation predictions than to make hard decisions for the users. For a given set of M predicted citations, we use a performance metric, Recall@ M ,

$$\text{Recall@}M = \frac{\text{number of citations in the predicted set}}{\text{total number of citations}}$$

which can be viewed as the proportion of “true” citations successfully predicted by a given method, when the method is allowed to provide M guesses.

Comparison methods. We compare our model with a number of competing strategies, starting with the RTM [9]. In order to make predictions using the RTM, we learn a latent representation for each document and predict citations using a similarity function between these representations (detailed in [9]). The second comparison is an LDA-based prediction strategy, in which document predictions are determined by the similarity between the latent document representation vectors θ_j . The similarity is computed using inverse of the Hellinger distance [6]

$$S_{ij} = H(\theta_i, \theta_j)^{-1} = \sqrt{2} \|\sqrt{\theta_i} - \sqrt{\theta_j}\|^{-1}.$$

Third, we compare with matrix factorization (MF), but only on the first task. (MF cannot make the citation predictions for a brand new document.) Finally, we compare with three simple baseline methods on both tasks. The first is that of Adamic/Adar [1], described in Section 4. The second is based on term frequency-inverse document frequency (TF-IDF) scores, where citations are predicted based on similarities in the documents’ scores [4]. The third baseline is called “in-degree”, where each document is given a score proportional to the number of times it is cited; in this case, the same set of predictions are given for every test document. Hyperparameters are set via cross validation.

Task one: predicting held-out citations.

Given the document contents and the remaining links, the task is to predict the held out citations for each document. We show results for our model and six comparison methods on the ACL dataset in Figure 3. Our model (LRO) achieves a significantly higher recall over all ranges of the number of predictions, and we observed similar results for the other two datasets.

We also wanted to determine how our method performs across different datasets. To make the results comparable, we normalized the number of predictions M by setting it to a fraction of the total number of documents in each respective dataset. The results are shown in Figure 4: LRO performs well on all three datasets, though we note that ACL has a much better score than

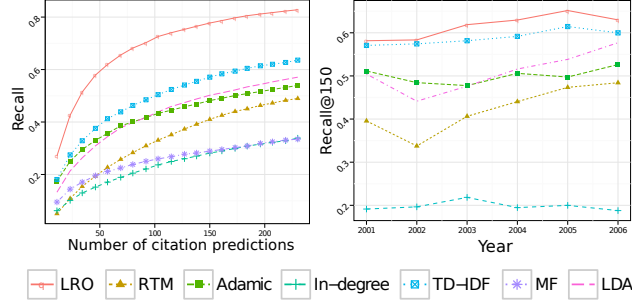


Figure 3: Left: Citation prediction performance on the ACL dataset for task one (predicting held-out citations). Right: Citation prediction performance on task two (predicting citations for new documents) on subsets of the ACL dataset for 7 years. In both cases, the LRO yields the highest recall over all ranges.

the other two. We attribute this to the fact that ACL contains only refereed academic papers, and is therefore more structured than either arXiv (which is unrefereed) or Simple English Wikipedia (whose articles are not always subject to editorial attention).

Task two: predicting citations for new documents. The second task is to predict citations for documents with no prior citation information, corresponding to scenarios in which one needs to suggest citations for newly written documents. This task is often referred to as the “cold start problem” in recommender systems.

We simulate the process of introducing newly written papers into a citation network by dividing them according to publication year. Specifically, from the ACL citation network dataset, we select the citations and documents that existed before the year Y as training data, for Y ranging from 2001 to 2006. After training on this subset, the task is then to predict the citations occurring in year Y for the new documents written in year Y .

For this task, we compared our model against the same comparison methods used in the previous task, except for matrix factorization, which cannot make citation predictions for new documents. Figure 3 (right) shows the results. We fix the number of citation predictions $M = 150$ (other M values have similar trends). Again, our model achieves the best performance over a majority of the M values in all six years, and increases its lead over the comparison methods in later years, after a larger portion of the citation network has formed and can be used as training data.

Hyperparameter sensitivity. We also study how different hyperparameters affect performance, including the number of topics K , precision parameters τ_0 and τ_1 , and latent random offset precision parameter λ

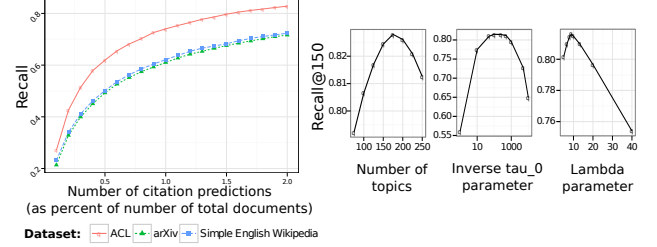


Figure 4: Left: citation prediction performance of our LRO model on three real-world datasets. The ACL dataset has a better score than the other two datasets. See main text for details. Right: citation prediction performance for a range of hyperparameter settings, including the number of topics K , the non-link variance parameter τ_0 , and the latent random offset variance parameter λ .

(Figure 4, right). Again, we fix $M = 150$. First, we varied the number of topics from 75 to 250, and found an optimal value of approximately 175 topics. Next, in order to find the optimal balance between parameters τ_0 and τ_1 , we fixed $\tau_1 = 1$ and varied τ_0 from $1/10000$ to 1, finding an optimal value of approximately $\tau_0 = 1/100$. Finally, we varied the parameter λ from 5 to 40, and found an optimal value at approximately $\lambda = 9$.

5.2 Exploring citation networks

The latent random offsets can yield useful information that allows for analysis and exploration of documents in the citation network. Our model provides, for each document, a content representation vector θ_j , which captures the topics associated with the content of the document, and a latent offset vector ϵ_j , which captures topics not necessarily contained within the document but expressed by others who cited the document. Highly positive latent offsets may capture the topics where a given document has been influential within the context of the citation network; alternatively, negative offsets can represent topics that are expressed highly in a document, but that have not proven to be influential within the context of the network.

Given a document, we can therefore explore its contents by examining the learned set of topics, and we can explore its role in the citation network (and see the topics of documents that it has influenced) by examining the latent offsets. In Figures 1 and 5 we show the latent topic representations of document contents, the learned random offsets, and the final augmented representations (the sum of topic representations and random offsets), for a document in each of the Simple English Wikipedia and ACL datasets. The augmented representations provide information on both the content and context of a document: they incorporate information contained in the document as well as in other

documents that cite it.

For highly cited documents, we have a great deal of information from the citing documents (i.e. the in-links), and this information can be used to more strongly offset the latent topic representations. Intuitively, the content is like a prior belief about a document’s latent representation, and as more sources start citing the document, this outside information further offsets the latent topic representations. Additionally, the offsets do not only “add” more information to the latent representation from the citing documents. In Figure 5 (top row), the offsets acted primarily to reduce the weights of many of the largest topics in the content representation, and only added weight to two topics. Here, the offsets served to dampen many of the content topics that did not appear to be relevant to the citing documents, and for this reason, the augmented representation is more sparse than the initial content representation.

Interpreting predictions. In addition to maintaining competitive prediction performance, our model allows for interpretable link prediction: for each predicted link we can use our latent representations to give users an understanding of why the link was returned. In particular, we can find the contribution that each topic provides to the final prediction score in order to determine the “reasons” (in terms of the latent topics) why a given document was predicted. We illustrate this in Figures 1 and 5 (bottom row of graphs). In Figure 1, for the *Sistine Chapel* document, *Chapel* is cited largely due to three topics (**architecture**, **Christianity**, and **buildings**), *Christian* is cited primarily due to a single topic (**Christianity**), and *Italy* is mainly cited due to six lower-weighted topics (**countries**, **Christianity**, **architecture**, **buildings**, **music**, and **populace**). Since *Italy* is a highly cited document and its augmented latent representation emphasizes a large number of topics (many of those expressed by its in-links), it was predicted due to a slight similarity in a number of topics as opposed to a strong similarity in just a few.

In Figure 5 we show three predictions for the document *Automatic Recognition of Chinese Unknown Words Based on Roles Tagging*. We can see that each of the predicted documents was due to a different aspect of this paper: the document *Automatic Rule Induction For Unknown-Word Guessing* was chosen primarily due to the **unknown-word** topic (related to the paper’s goal of recognizing unknown words), the document *Word Identification for Mandarin Chinese Sentences* was chosen primarily due to the **China** topic (related to the paper’s language domain area), and the document *A Knowledge-Free Method For Capitalized Word Disambiguation* was chosen primarily due to the **pronoun** topic (related to the paper’s use of names, locations,

and roles).

Automatic Recognition Of Chinese Unknown Words Based On Roles Tagging (ACL)

Text: “This paper ... is based on the idea of ‘roles tagging’, to the complicated problems of Chinese unknown words recognition ... an unknown word is identified according to its component tokens and context tokens. In order to capture the functions of tokens, we use the concept of roles...We have got excellent precision and recalling rates, especially for person names and transliterations...”

In-Links (Citing Documents): (1) A...word segmentation system for Chinese, (2) Chinese lexical analysis..., (3) HHMM-based Chinese lexical analyzer..., (4) Chinese word segmentation...of characters, (5) Chinese unknown...character-based tagging...

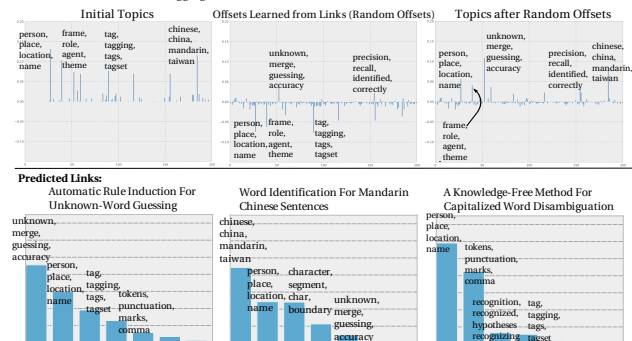


Figure 5: Interpreting citation predictions for the document *Automatic Recognition Of Chinese Unknown Words Based On Roles Tagging* in the ACL dataset. For each predicted link, we show the relative weight that each latent topic (denoted by the top four words) contributed to the prediction. These provide reasons why each predicted link was chosen, in terms of the topics.

6 Conclusion

In this paper, we proposed a probabilistic approach for citation network modeling that integrates the merits of both content and link analyses. Our empirical results showed improved performance compared with several popular approaches for citation prediction. Furthermore, our approach can suggest citations for brand new documents without prior citations—an essential ability for building a real citation recommendation system.

Qualitatively, our approach provides meaningful explanations for how predictions are made, through the latent random offsets. These explanations provide additional information that can be useful for making informed decisions. For example, in a citation recommendation system, we can inform users whether a citation is suggested more due to content similarities or due to the existing network structure, and we can show the relative amounts that individual topics contributed to the prediction. In future work, we would like to conduct user studies to quantify how this additional information helps users find more relevant citations in a more efficient way.

7 Acknowledgments

This work is supported in part by DARPA X Data FA87501220324, NIH GWAS R01GM087694, and NSF Social Media IIS1111142. Q. Ho is supported by an A-STAR, Singapore fellowship.

References

- [1] Lada A Adamic and Eytan Adar, *Friends and neighbors on the web*, Social networks **25** (2003), no. 3, 211–230.
- [2] Edoardo M Airoldi, David M Blei, Stephen E Fienberg, and Eric P Xing, *Mixed membership stochastic block-models*, The Journal of Machine Learning Research **9** (2008), 1981–2014.
- [3] D. Bertsekas, *Nonlinear programming*, Athena Scientific, 1999.
- [4] Steven Bethard and Dan Jurafsky, *Who should I cite: learning literature search models from citation behavior*, International Conference on Information and Knowledge Management, 2010, pp. 609–618.
- [5] D. Blei, *Probabilistic topic models*, Communications of the ACM **55** (2012), no. 4, 77–84.
- [6] D. Blei and J. Lafferty, *Topic models*, Text Mining: Theory and Applications (A. Srivastava and M. Sahami, eds.), Taylor and Francis, 2009.
- [7] D. Blei, A. Ng, and M. Jordan, *Latent Dirichlet allocation*, Journal of Machine Learning Research **3** (2003), 993–1022.
- [8] Allison June-Barlow Chaney and David M. Blei, *Visualizing topic models*, The International AAAI Conference on Weblogs and Social Media, 2012.
- [9] J. Chang and D. Blei, *Relational topic models for document networks*, Artificial Intelligence and Statistics, 2009.
- [10] J. Chang, J. Boyd-Graber, C. Wang, S. Gerrish, and D. Blei, *Reading tea leaves: How humans interpret topic models*, Advances in Neural Information Processing Systems (NIPS), 2009.
- [11] Qirong Ho, Jacob Eisenstein, and Eric P Xing, *Document hierarchies from text and links*, Proceedings of the 21st international conference on World Wide Web, ACM, 2012, pp. 739–748.
- [12] M. Hoffman, D. Blei, C. Wang, and J. Paisley, *Stochastic Variational Inference*, ArXiv e-prints (2012).
- [13] Yifan Hu, Yehuda Koren, and Chris Volinsky, *Collaborative filtering for implicit feedback datasets*, IEEE International Conference on Data Mining, 2008.
- [14] Leo Katz, *A new status index derived from sociometric analysis*, Psychometrika **18** (1953), no. 1, 39–43.
- [15] Yehuda Koren, Robert Bell, and Chris Volinsky, *Matrix factorization techniques for recommender systems*, Computer **42** (2009), no. 8, 30–37.
- [16] Yan Liu, Alexandru Niculescu-Mizil, and Wojciech Gryc, *Topic-link LDA: joint models of topic and author community*, International Conference on Machine Learning, ACM, 2009, pp. 665–672.
- [17] Ramesh Nallapati and William Cohen, *Link-PLSA-LDA: A new unsupervised model for topics and influence of blogs*, International Conference for Weblogs and Social Media, 2008.
- [18] Ramesh Nallapati, Daniel A Mcfarland, and Christopher D Manning, *Topicflow model: Unsupervised learning of topic-specific influences of hyperlinked documents*, International Conference on Artificial Intelligence and Statistics, 2011, pp. 543–551.
- [19] Mark EJ Newman, *Modularity and community structure in networks*, Proceedings of the National Academy of Sciences **103** (2006), no. 23, 8577–8582.
- [20] Taeshik Shon and Jongsu Moon, *A hybrid machine learning approach to network anomaly detection*, Information Sciences **177** (2007), no. 18, 3799–3821.
- [21] Alexander Smola and Shravan Narayanamurthy, *An architecture for parallel topic models*, Proc. VLDB Endow. **3** (2010), no. 1-2, 703–710.
- [22] Chong Wang and David Blei, *Collaborative topic modeling for recommending scientific articles.*, ACM International Conference on Knowledge Discovery and Data Mining (KDD), 2011.
- [23] X. Wei and B. Croft, *LDA-based document models for ad-hoc retrieval*, SIGIR, 2006.

Collaborative Multi-output Gaussian Processes

Trung V. Nguyen
ANU & NICTA
Canberra, Australia

Edwin V. Bonilla
NICTA & ANU
Sydney, Australia

Abstract

We introduce the collaborative multi-output Gaussian process (GP) model for learning dependent tasks with very large datasets. The model fosters task correlations by mixing sparse processes and sharing multiple sets of inducing points. This facilitates the application of variational inference and the derivation of an evidence lower bound that decomposes across inputs and outputs. We learn all the parameters of the model in a single stochastic optimization framework that scales to a large number of observations per output and a large number of outputs. We demonstrate our approach on a toy problem, two medium-sized datasets and a large dataset. The model achieves superior performance compared to single output learning and previous multi-output GP models, confirming the benefits of correlating sparsity structure of the outputs via the inducing points.

1 INTRODUCTION

Gaussian process models (GPs, Rasmussen and Williams, 2006) are a popular choice in Bayesian regression due to their ability to capture complex dependencies and non-linearities in data. In particular, when having multiple outputs or tasks they have proved effective in modeling the dependencies between the tasks, outperforming competitive baselines and single output learners (Bonilla et al., 2008; Teh et al., 2005; Alvarez and Lawrence, 2009; Wilson et al., 2012). However, the prohibitive cost of performing exact inference in GP models severely hinders their application to large scale multi-output problems. For example, naïve inference in a fully coupled Gaussian process model over P outputs and N data points can have

a complexity of $\mathcal{O}(N^3P^3)$ and $\mathcal{O}(N^2P^2)$ in time and memory, respectively.

A motivating example of a large scale multi-output application is the tracking of movements of a robot arm using 2 or more joint torques. If one of the robot motors malfunctions and fails to record a torque, data collected from the other motors may be used to infer the missing torque values. However, taking 100 measurements per second already results in a total of over 40,000 data points per torque in just 7 minutes. Clearly this problem is well beyond the capabilities of conventional multiple output GPs. Building multi-output GP models that can learn correlated tasks at the scale of these types of problems is thus the main focus of this paper.

In the single output setting previous attempts to scale up GP inference resort to approximate inference. Most approximation methods can be understood within a single probabilistic framework that uses a set of *inducing points* in order to obtain an approximate process (or a low-rank approximate covariance) over which inference can be performed more efficiently (Quiñero-Candela and Rasmussen, 2005). These models have been referred to in the literature as sparse models. Nevertheless, straightforward application of such approximate techniques will yield a computational cost of at least $\mathcal{O}(PNM^2)$ in time and $\mathcal{O}(PNM)$ in memory, where M is the number of inducing points. This high complexity still prevents us from applying GP models to large scale multi-output problems.

In this work we approach the challenge of building scalable multi-output Gaussian process models based on the following observations. Firstly, inducing variables are the key catalyst for achieving sparsity and dealing with large scale problems in Gaussian process models. In particular, they capture the *sufficient statistics* of a dataset allowing the construction of sparse processes that can approximate arbitrarily well the exact GP model (Titsias, 2009). Secondly, the use of global latent variables (such as the inducing points) allows us

to induce dependencies in a highly correlated model efficiently. This observation is exploited in Hensman et al. (2013) for single output GP regression models where, by explicitly representing a distribution over the inducing variables, stochastic variational inference can be used to work with millions of data points. Finally, the key to multi-output and multi-task learning is to model dependencies between the outputs based on realistic assumptions of what can be shared across the tasks. It turns out that sharing “sparsity structure” can not only be a reasonable assumption but also a crucial component when modeling dependencies between different related tasks.

Based on these observations, we propose the collaborative multi-output Gaussian Process (COGP) model where latent processes are mixed to generate dependent outputs. Each process is sparse and characterized by its own set of inducing points. The sparsity structure enabling output correlations is thus created via the shared inducing sets. To learn this structure, we maintain an explicit representation of the posterior over the inducing points which in turn allows inference to be carried out efficiently. In particular, we obtain a variational lower bound of the model evidence that decomposes across inputs and outputs. This decomposition makes possible the application of stochastic variational inference, thus allowing the model to handle a large number of observations per output and a large number of outputs. Furthermore, learning of all the parameters in the model, including kernel hyperparameters and inducing inputs, can be done in a single stochastic optimization framework.

We analyze our multi-out model on a toy problem where the inducing variables are shown to be conducive to the sharing of information between two related tasks. Additionally, we evaluate our model on two moderate-sized datasets in which we show that it can outperform previous non-scalable multi-output approaches as well as single output baselines. Finally, on a large scale experiment regarding the learning of robot inverse dynamics we show the substantial benefits of collaborative learning provided by our model.

Related work. Most GP-based multi-output models create correlated outputs by mixing a set of independent latent processes. The mixing can be a linear combination with fixed coefficients (see e.g. Teh et al., 2005; Bonilla et al., 2008). This is known in the geostatistics community as the “linear model of coregionalization” (Goovaerts, 1997). Such models may also be reformulated in a common Bayesian framework, for example by placing a spike and slab prior over the coefficients (Titsias and Lázaro-Gredilla, 2011). More complex dependencies can be induced

by using input-dependent coefficients (Wilson et al., 2012; Nguyen and Bonilla, 2013) or convolving processes (Boyle and Frean, 2005; Alvarez and Lawrence, 2009; Alvarez et al., 2010).

While we also use the mixing construction, the key difference in our model is the role of inducing variables. In particular, when used in previous models to reduce the computational costs (see e.g. Alvarez and Lawrence, 2009; Álvarez et al., 2010), the inducing points are integrated out or collapsed. In contrast, our model maintains an explicit representation of the posterior over the inducing variables that is learned using data from all outputs. This explicit representation facilitates scalable learning in a similar fashion to the approach in Hensman et al. (2013), making it applicable to very large datasets.

2 MODEL SPECIFICATION

Before diving into technical details of the model specification, we discuss the modeling philosophy behind our collaborative multi-output Gaussian processes. To learn the outputs jointly, we need a mechanism through which information can be transferred among the outputs. This is achieved in the model by allowing the outputs to share multiple sets of inducing variables, each of which captures a different pattern common to the outputs. These variables play a double pivotal role in the model: they collaboratively share information across the outputs and provide sufficient statistics so as to induce sparse processes.

Consider the joint regression of P tasks with inputs $\mathbf{X} = \{\mathbf{x}_n \in \mathcal{R}^D\}_{n=1}^N$ and outputs $\mathbf{y} = \{\mathbf{y}_i\}_{i=1}^P$ where $\mathbf{y}_i = \{y_{in}\}_{n=1}^N$. We model each output as a weighted combination of Q shared latent functions $\{g_j\}_{j=1}^Q$, plus an individual latent function $\{h_i\}_{i=1}^P$ unique to that output for greater flexibility. The Q shared functions have independent Gaussian process priors $g_j(\mathbf{x}) \sim \mathcal{GP}(0, k_j(\cdot, \cdot))$. Similarly, each individual function of an output also has a GP prior, i.e. $h_i(\mathbf{x}) \sim \mathcal{GP}(0, k_i^h(\cdot, \cdot))$.

As we want to sparsify these processes, we introduce a set of *shared inducing variables* \mathbf{u}_j for each $g_j(\mathbf{x})$, i.e. \mathbf{u}_j contains the values of $g_j(\mathbf{x})$ at the inducing inputs \mathbf{Z}_j . Likewise, we have individual inducing variables corresponding to each $h_i(\mathbf{x})$, which we denote with \mathbf{v}_i and their corresponding inducing inputs \mathbf{Z}_i^h . The inducing inputs lie in the same space as the inputs \mathbf{X} . For convenience, we assume all processes have the same number of inducing points, M . However we emphasize that this is not imposed in practice.

We denote the collective variables: $\mathbf{g} = \{\mathbf{g}_j\}$, $\mathbf{h} = \{\mathbf{h}_i\}$, $\mathbf{u} = \{\mathbf{u}_j\}$, $\mathbf{v} = \{\mathbf{v}_i\}$, $\mathbf{Z} = \{\mathbf{Z}_j\}$, and $\mathbf{Z}^h = \{\mathbf{Z}_i^h\}$

where $\mathbf{g}_j = \{g_j(\mathbf{x}_n)\}$, $\mathbf{h}_i = \{h_i(\mathbf{x}_n)\}$. Note that we reserve subscript i for indexing the outputs and their corresponding individual processes ($i = 1 \dots P$), j for the shared latent processes ($j = 1 \dots Q$), and n for the inputs ($n = 1 \dots N$).

2.1 PRIOR MODEL

From the definition of the GPs and the independence of the processes, the *prior* of the multi-output model can be written as:

$$p(\mathbf{g}|\mathbf{u}) = \prod_{j=1}^Q p(\mathbf{g}_j|\mathbf{u}_j) = \prod_{j=1}^Q \mathcal{N}(\mathbf{g}_j; \boldsymbol{\mu}_j, \tilde{\mathbf{K}}_j) \quad (1)$$

$$p(\mathbf{u}) = \prod_{j=1}^Q p(\mathbf{u}_j) = \prod_{j=1}^Q \mathcal{N}(\mathbf{u}_j; \mathbf{0}, k(\mathbf{Z}_j, \mathbf{Z}_j)) \quad (2)$$

$$p(\mathbf{h}|\mathbf{v}) = \prod_{i=1}^P p(\mathbf{h}_i|\mathbf{v}_i) = \prod_{i=1}^P \mathcal{N}(\mathbf{h}_i; \boldsymbol{\mu}_i^h, \tilde{\mathbf{K}}_i^h) \quad (3)$$

$$p(\mathbf{v}) = \prod_{i=1}^P p(\mathbf{v}_i) = \prod_{i=1}^P \mathcal{N}(\mathbf{v}_i; \mathbf{0}, k(\mathbf{Z}_i^h, \mathbf{Z}_i^h)), \quad (4)$$

where the corresponding means and covariances of the Gaussians are given by:

$$\boldsymbol{\mu}_j = k(\mathbf{X}, \mathbf{Z}_j)k(\mathbf{Z}_j, \mathbf{Z}_j)^{-1}\mathbf{u}_j \quad (5)$$

$$\boldsymbol{\mu}_i^h = k(\mathbf{X}, \mathbf{Z}_i^h)k(\mathbf{Z}_i^h, \mathbf{Z}_i^h)^{-1}\mathbf{v}_i \quad (6)$$

$$\tilde{\mathbf{K}}_j = k_j(\mathbf{X}, \mathbf{X}) - k(\mathbf{X}, \mathbf{Z}_j)k(\mathbf{Z}_j, \mathbf{Z}_j)^{-1}k(\mathbf{Z}_j, \mathbf{X}) \quad (7)$$

$$\tilde{\mathbf{K}}_i^h = k_i^h(\mathbf{X}, \mathbf{X}) - k(\mathbf{X}, \mathbf{Z}_i^h)k(\mathbf{Z}_i^h, \mathbf{Z}_i^h)^{-1}k(\mathbf{Z}_i^h, \mathbf{X}). \quad (8)$$

In the equations and hereafter, we omit the subscripts j, h, i from the kernels $k_j(\cdot, \cdot)$ and $k_i^h(\cdot, \cdot)$ when it is clear from the parameters inside the parentheses which covariance function is in action.

Equations (2) and (4) follow directly from the properties of GPs, while the expressions for $p(\mathbf{g}|\mathbf{u})$ and $p(\mathbf{h}|\mathbf{v})$ (Equations (1) and (3)) come from the conditionals of the multivariate Gaussian distributions. Instead of writing the joint priors $p(\mathbf{g}, \mathbf{u})$ and $p(\mathbf{h}, \mathbf{v})$, the above equivalent equations are given to emphasize the sufficient statistics role of \mathbf{u} and \mathbf{v} in the model. Here by sufficient statistics we mean, for any sparse process (say g_j), any other set of function values is independent of \mathbf{g}_j given the inducing variables \mathbf{u}_j .

2.2 LIKELIHOOD MODEL

As mentioned above, we assume that observations for each output are (noisy) linear combinations of the Q latent functions $g_j(\mathbf{x})$ plus an independent function $h_i(\mathbf{x})$. Hence we have that the likelihood with stan-

dard iid Gaussian noise is given by:

$$p(\mathbf{y}|\mathbf{g}, \mathbf{h}) = \prod_{i=1}^P \prod_{n=1}^N \mathcal{N}(y_{in}; \sum_{j=1}^Q w_{ij}g_j(\mathbf{x}_n) + h_i(\mathbf{x}_n), \beta_i^{-1}), \quad (9)$$

where w_{ij} are the corresponding weights and β_i is the precision of each Gaussian. As the latent values \mathbf{g} are specified conditioned on the inducing variables \mathbf{u} , this construction implies that each output is a weighted combination of the inducing values. We note that if \mathbf{u} and \mathbf{v} are marginalized out, we obtain the semiparametric latent factor model (Teh et al., 2005). However, doing so is against the purpose of our model which encourages sharing of outputs via the inducing variables. Furthermore, as we shall see in the next section, explicit representation of these variables is fundamental to scalable inference of the model.

3 INFERENCE

We approximate the posterior over the latent variables $\mathbf{g}, \mathbf{h}, \mathbf{u}, \mathbf{v}$ given observations \mathbf{y} using variational inference (Jordan et al., 1999). In section 3.1 we derive a lower bound of the marginal likelihood which has the key property of factorizing over the data points and outputs. Section 3.2 takes advantage of this factorization to derive stochastic variational inference, allowing the model to scale to very large datasets. Section 3.3 compares the complexity of the model with previous multi-output methods.

3.1 VARIATIONAL LOWER BOUND

In variational inference, we find the “closest” approximate distribution to the true posterior in terms of the KL divergence. We first observe that the true posterior distribution can be written as:

$$p(\mathbf{g}, \mathbf{h}, \mathbf{u}, \mathbf{v}|\mathbf{y}) = p(\mathbf{g}|\mathbf{u}, \mathbf{y})p(\mathbf{h}|\mathbf{v}, \mathbf{y})p(\mathbf{u}, \mathbf{v}|\mathbf{y}). \quad (10)$$

Here we recall the modeling assumption that each set of inducing variables is the sufficient statistics of the corresponding latent process. This motivates replacing the true posteriors over \mathbf{g} and \mathbf{h} with their conditional distributions given the inducing variables, leading to a distribution of the form:

$$q(\mathbf{g}, \mathbf{h}, \mathbf{u}, \mathbf{v}|\mathbf{y}) = p(\mathbf{g}|\mathbf{u})p(\mathbf{h}|\mathbf{v})q(\mathbf{u}, \mathbf{v}), \quad (11)$$

with

$$q(\mathbf{u}, \mathbf{v}) = \prod_{j=1}^Q \underbrace{\mathcal{N}(\mathbf{u}_j; \mathbf{m}_j, \mathbf{S}_j)}_{q(\mathbf{u}_j)} \prod_{i=1}^P \underbrace{\mathcal{N}(\mathbf{v}_i; \mathbf{m}_i^h, \mathbf{S}_i^h)}_{q(\mathbf{v}_i)}. \quad (12)$$

This technique has been used by Titsias (2009) and Hensman et al. (2013) to derive variational inference algorithms for the single output case. Since the conditionals $p(\mathbf{g}|\mathbf{u})$ and $p(\mathbf{h}|\mathbf{v})$ are known (Equations (1) and (3)), we only need to learn $q(\mathbf{u}, \mathbf{v})$ so as to minimize the divergence between the approximate posterior and the true posteriors. The quality of approximation depends entirely on the posterior over the inducing variables, thus underlining their pivotal role in the model as previously discussed.

To find the best $q(\mathbf{u}, \mathbf{v})$, we optimize the evidence lower bound (ELBO) of the log marginal:

$$\log p(\mathbf{y}) \geq \int q(\mathbf{u}, \mathbf{v}) \log p(\mathbf{y}|\mathbf{u}, \mathbf{v}) d\mathbf{u} d\mathbf{v} - \sum_{j=1}^Q \text{KL}[q(\mathbf{u}_j)||p(\mathbf{u}_j)] - \sum_{i=1}^P \text{KL}[q(\mathbf{v}_i)||p(\mathbf{v}_i)], \quad (13)$$

which is derived using Jensen's inequality and the fact that both of $q(\mathbf{u}, \mathbf{v})$ and $p(\mathbf{u}, \mathbf{v})$ fully factorize. Since $q(\mathbf{u}_j), q(\mathbf{v}_i), p(\mathbf{u}_j), p(\mathbf{v}_i)$ are all multivariate Gaussian distributions, the KL divergence terms are analytically tractable. To compute the expected likelihood term in the ELBO we first see that

$$\log p(\mathbf{y}|\mathbf{u}, \mathbf{v}) \geq \langle \log p(\mathbf{y}|\mathbf{g}, \mathbf{h}) \rangle_{p(\mathbf{g}, \mathbf{h}|\mathbf{u}, \mathbf{v})} = \sum_{i=1}^P \sum_{n=1}^N \langle \log p(y_{in}|\mathbf{g}_n, h_{in}) \rangle_{p(\mathbf{g}|\mathbf{u})p(\mathbf{h}_i|\mathbf{v}_i)} \quad (14)$$

where $\mathbf{g}_n = \{g_{jn} = (\mathbf{g}_j)_n\}_{j=1}^Q$. The inequality is due to Jensen's inequality and the equality is due to the factorization of the likelihood.

The ELBO can be computed by first solving for the individual expectations $\langle \log p(y_{in}|\mathbf{g}_n, h_{in}) \rangle$ over $p(\mathbf{g}|\mathbf{u})p(\mathbf{h}_i|\mathbf{v}_i)$ and then substituting these into Equation (13) (see the supplementary material for details). Hence the resulting lower bound is given by:

$$\begin{aligned} \mathcal{L} = & \sum_{i,n} \left(\log \mathcal{N}(y_{in}; \tilde{\mu}_{in}, \beta_i^{-1}) - \frac{1}{2} \beta_i \sum_{j=1}^Q w_{ij}^2 \tilde{k}_{jnn} \right. \\ & \left. - \frac{1}{2} \beta_i \tilde{k}_{inn}^h - \frac{1}{2} \beta_i \sum_{j=1}^Q \text{tr } w_{ij}^2 \mathbf{S}_j \mathbf{\Lambda}_{jn} - \beta_i \frac{1}{2} \text{tr } \mathbf{S}_i^h \mathbf{\Lambda}_{in} \right) \\ & - \sum_{j=1}^Q \left(\frac{1}{2} \log |\mathbf{K}_{jzz} \mathbf{S}_j^{-1}| + \frac{1}{2} \text{tr } \mathbf{K}_{jzz}^{-1} (\mathbf{m}_j \mathbf{m}_j^T + \mathbf{S}_j) \right) \\ & - \sum_{i=1}^P \left(\frac{1}{2} \log |\mathbf{K}_{izz} (\mathbf{S}_i^h)^{-1}| \right. \\ & \left. + \frac{1}{2} \text{tr } \mathbf{K}_{izz}^{-1} (\mathbf{m}_i^h (\mathbf{m}_i^h)^T + \mathbf{S}_i^h) \right), \quad (15) \end{aligned}$$

where $\mathbf{K}_{jzz} = k(\mathbf{Z}_j, \mathbf{Z}_j)$, $\mathbf{K}_{izz} = k(\mathbf{Z}_i^h, \mathbf{Z}_i^h)$, and:

$$\tilde{\mu}_{in} = \sum_{j=1}^Q w_{ij} \mathbf{A}_j(n, :) \mathbf{m}_j + \mathbf{A}_i^h(n, :) \mathbf{m}_i^h, \quad (16)$$

$$\mathbf{\Lambda}_{jn} = \mathbf{A}_j(n, :)^T \mathbf{A}_j(n, :), \quad (17)$$

$$\mathbf{\Lambda}_{in} = \mathbf{A}_i^h(n, :)^T \mathbf{A}_i^h(n, :), \quad (18)$$

with $\tilde{k}_{jnn} = (\tilde{\mathbf{K}}_j)_{nn}$; $\tilde{k}_{inn}^h = (\tilde{\mathbf{K}}_i^h)_{nn}$; $\mu_{jn} = (\boldsymbol{\mu}_j)_n$; $\mu_{in}^h = (\boldsymbol{\mu}_i^h)_n$; and we have defined the auxiliary matrices $\mathbf{A}_j = k(\mathbf{X}, \mathbf{Z}_j) \mathbf{K}_{jzz}^{-1}$ and $\mathbf{A}_i^h = k(\mathbf{X}_i, \mathbf{Z}_i^h) \mathbf{K}_{izz}^{-1}$ and used $\mathbf{A}_j(n, :)$ to denote the n -th row vector of \mathbf{A}_j . Notice that this ELBO generalizes the bound for standard GP regression derived in Hensman et al. (2013), which can be recovered by setting $P = Q = 1$, $w_{ij} = 1$ and $h_i(\mathbf{x}) = 0$.

The novelty of the variational lower bound in Equation (15) is that it decomposes across both inputs and outputs. This enables the use of stochastic optimization methods, which allow the model to handle very large datasets for which existing GP-based multi-output models are simply impractical.

3.2 STOCHASTIC VARIATIONAL INFERENCE

So far in the description of the model and inference we have implicitly assumed that every output has full observations at all inputs \mathbf{X} . To discern where learning occurs for each output, we make the missing data scenario more explicit. Specifically, each output i can have observations at a different set of inputs \mathbf{X}_i . We shall use \mathbf{o}_i to denote the indices of \mathbf{X}_i (in the set \mathbf{X}) and use the indexing operator $\mathbf{B}(\mathbf{o}_i)$ to select the rows corresponding to \mathbf{o}_i from any arbitrary matrix \mathbf{B} . We also overload \mathbf{y}_i as the observed targets of output i .

3.2.1 Learning the Parameters of the Variational Distribution

We can obtain the derivatives of the ELBO in Equation (15) wrt the variational parameters for optimization. The derivatives of \mathcal{L} wrt the parameters of $q(\mathbf{u}_j)$ are given by:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{m}_j} = \sum_{i=1}^P \beta_i w_{ij} \mathbf{A}_j(\mathbf{o}_i)^T \mathbf{y}_i^{\setminus j} \quad (19)$$

$$- \left[\mathbf{K}_{jzz}^{-1} + \sum_{i=1}^P \beta_i w_{ij}^2 \mathbf{A}_j(\mathbf{o}_i)^T \mathbf{A}_j(\mathbf{o}_i) \right] \mathbf{m}_j,$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{S}_j} = \frac{1}{2} \mathbf{S}_j^{-1} - \frac{1}{2} \left[\mathbf{K}_{jzz}^{-1} + \sum_{i=1}^P \beta_i w_{ij}^2 \mathbf{A}_j(\mathbf{o}_i)^T \mathbf{A}_j(\mathbf{o}_i) \right], \quad (20)$$

where $\mathbf{y}_i^{\setminus j} = \mathbf{y}_i - \mathbf{A}_i^h(\mathbf{o}_i) \mathbf{m}_i^h - \sum_{j' \neq j} w_{ij'} \mathbf{A}_{j'}(\mathbf{o}_i) \mathbf{m}_{j'}$.

The derivatives of \mathcal{L} wrt the parameters of $q(\mathbf{v}_i)$ are given by:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{m}_i^h} = \beta_i \mathbf{A}_i^h(\mathbf{o}_i)^T \mathbf{y}_i^{\setminus h} - \left[\mathbf{K}_{izz}^{-1} + \beta_i \mathbf{A}_i^h(\mathbf{o}_i)^T \mathbf{A}_i^h(\mathbf{o}_i) \right] \mathbf{m}_i, \quad (21)$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{S}_i^h} = \frac{1}{2} \mathbf{S}_i^{-1} - \frac{1}{2} \left[\mathbf{K}_{izz}^{-1} + \beta_i \mathbf{A}_i^h(\mathbf{o}_i)^T \mathbf{A}_i^h(\mathbf{o}_i) \right], \quad (22)$$

where $\mathbf{y}_i^{\setminus h} = \mathbf{y}_i - \sum_{j=1}^Q w_{ij} \mathbf{A}_j(\mathbf{o}_i, :) \mathbf{m}_j$.

It can be seen that the derivatives of the parameters of $q(\mathbf{v}_i)$ only involve the observations of the output i . The derivatives of the parameters of $q(\mathbf{u}_j)$ involve the observations of all outputs but is a sum of contributions from individual outputs. Computation of the derivatives can therefore be easily distributed or parallelized.

Since the optimal distributions $q(\mathbf{u}_j)$ and $q(\mathbf{v}_i)$ are in the exponential family, it is more convenient to use stochastic variational inference (Hensman et al., 2012, 2013) to perform update of their canonical parameters. This works by taking a step of length l in the direction of the natural gradient approximated by mini-batches of the data. For instance, consider $q(\mathbf{u}_j)$ whose canonical parameters are $\Phi_1 = \mathbf{S}_j^{-1} \mathbf{m}_j$ and $\Phi_2 = -\frac{1}{2} \mathbf{S}_j^{-1}$. Their stochastic update equations at time $t + 1$ are given by:

$$\Phi_{1(t+1)} = \mathbf{S}_{j(t)}^{-1} \mathbf{m}_{j(t)} + l \left(\sum_{i=1}^P \beta_i w_{ij} \mathbf{A}_j(\mathbf{o}_i)^T \mathbf{y}_i^{\setminus j} - \mathbf{S}_{j(t)}^{-1} \mathbf{m}_{j(t)} \right) \quad (23)$$

$$\Phi_{2(t+1)} = -\frac{1}{2} \mathbf{S}_{j(t)}^{-1} + l \left(\frac{1}{2} \mathbf{S}_{j(t)}^{-1} - \frac{1}{2} \mathbf{\Lambda} \right), \quad (24)$$

where $\mathbf{\Lambda} = \mathbf{K}_{jzz}^{-1} + \sum_{i=1}^P \beta_i w_{ij}^2 \mathbf{A}_j(\mathbf{o}_i)^T \mathbf{A}_j(\mathbf{o}_i)$.

3.2.2 Inducing Inputs and Hyper-parameters

To learn the hyperparameters, which in this model include the mixing weights, the covariance hyperparameters of the latent processes, and the noise precision of each output, we follow standard practice in GP inference. For this model this involves taking derivatives of the ELBO and applying standard stochastic gradient descent in alternative steps with the variational parameters, much like a variational EM algorithm. The derivatives are given in the supplementary material.

Learning of the inducing inputs, which was not considered in the single output case in Hensman et al. (2013), is also possible in our stochastic optimization approach. In the supplementary material, we show

Table 1: Comparison of the time and storage complexity of approximate inference of multi-output GP models. A&W, 2009 refers to Alvarez and Lawrence (2009). COGP is the only method with complexity *independent* of the number of inputs N and outputs P , thus it can scale to very large datasets.

METHOD	TIME	STORAGE
COGP, this paper	$\mathcal{O}(M^3)$	$\mathcal{O}(M^2)$
SLFM, (Teh et al., 2005)	$\mathcal{O}(QNM_t^2)$	$\mathcal{O}(QNM_t)$
MTGP, (Bonilla et al., 2008)	$\mathcal{O}(PNM_t^2)$	$\mathcal{O}(PNM_t)$
CGP-FITC (A&W, 2009)	$\mathcal{O}(PNM_t^2)$	$\mathcal{O}(PNM_t)$
GPRN, (Wilson et al., 2012)	$\mathcal{O}(PQN^3)$	$\mathcal{O}(PQN^2)$

that the additional cost of computing the derivatives of the lower bound wrt the inducing inputs is not significantly higher than the cost of updating the variational parameters. This makes optimizing the inducing locations a practical option, which can be critical in high-dimensional problems. Indeed, our experiments on a large scale multi-output problem show that automatic learning of the inducing inputs can lead to significant performance gain with little overhead in computation.

3.3 COMPLEXITY ANALYSIS

In this section we analyze the complexity of the model and compare it to existing multi-output approaches. For consistency, we first unify common notations used for all models. We use P as the number of outputs; N as the number of inputs; Q as the number of shared latent processes; and M_t as the *total* number of inducing inputs. It is worth noting that $M_t = (P + Q) \times M$ in our COGP model, assuming that each sparse process has equal number of inducing points. Also, COGP has P additional individual processes, one for each output.

The complexity of COGP can be read off by inspecting the ELBO in Equation (15), with the key observation that it contains a sum over the outputs as well as over the inputs. This means a mini-batch containing a small subset of the inputs and outputs can be used for stochastic optimization. Technically, the cost is $\mathcal{O}(M^3)$ or $\mathcal{O}(N_b M^2)$, where N_b is the size of the mini-batches, depending on which is larger between M and N_b . In practice, we may use $N_b > M$ as a large batch size (e.g. $N_b = 1000$) helps reduce stochasticity of the optimization. However, here we use $\mathcal{O}(M^3)$ for easier comparison with other models whose time and storage demands are given in Table 1. We see that COGP has a computational complexity that is independent of the size of the inputs and outputs, which makes it the only method capable of handling large scale problems.

3.4 PREDICTION

The predictive distribution of the i -th output for a test input \mathbf{x}_* is given by:

$$p(f_*|\mathbf{y}, \mathbf{x}_*) = \mathcal{N}(f_*; \sum_{j=1}^Q w_{ij} \mu_{j*} + \mu_{i*}^h, w_{ij}^2 s_{j*} + s_{i*}^h), \quad (25)$$

where μ_{j*} and s_{j*} are the mean and variance of the prediction for $g_{j*} = g_j(\mathbf{x}_*)$, i.e. $p(g_{j*}|\mathbf{y}, \mathbf{x}_*) = \mathcal{N}(g_{j*}; \mu_{j*}, s_{j*})$. Likewise, μ_{i*}^h and s_{i*}^h are the mean and variance of the prediction for $h_{i*} = h_i(\mathbf{x}_*)$, $p(h_{i*}|\mathbf{y}, \mathbf{x}_*) = \mathcal{N}(h_{i*}; \mu_{i*}^h, s_{i*}^h)$. These predictive means and variances are given by:

$$\mu_{j*} = \mathbf{k}_{j*z} \mathbf{K}_{jzz}^{-1} \mathbf{m}_j, \quad (26)$$

$$s_{j*} = k_{j**} - \mathbf{k}_{j*z} (\mathbf{K}_{jzz}^{-1} - \mathbf{K}_{jzz}^{-1} \mathbf{S}_j \mathbf{K}_{jzz}^{-1}) \mathbf{k}_{j*z}^T, \quad (27)$$

$$\mu_{i*}^h = \mathbf{k}_{i*z} \mathbf{K}_{izz}^{-1} \mathbf{m}_i^h, \quad (28)$$

$$s_{i*}^h = k_{i**} - \mathbf{k}_{i*z} (\mathbf{K}_{izz}^{-1} - \mathbf{K}_{izz}^{-1} \mathbf{S}_i \mathbf{K}_{izz}^{-1}) \mathbf{k}_{i*z}^T, \quad (29)$$

where $k_{j**} = k_j(\mathbf{x}_*, \mathbf{x}_*)$, $k_{i**} = k_i^h(\mathbf{x}_*, \mathbf{x}_*)$, \mathbf{k}_{j*z} is the covariance between \mathbf{x}_* and \mathbf{Z}_j , and \mathbf{k}_{i*z} is the covariance between \mathbf{x}_* and \mathbf{Z}_i^h .

4 EXPERIMENTS

We evaluate the proposed approach with four experiments. A toy problem is first used to study the transfer of learning between two related processes via the shared inducing points. We then compare the model with existing multi-output models on the tasks of predicting foreign exchange rate and air temperature. In the final experiment, we show that joint learning under sparsity can yield significant performance gain on a large scale dataset of inverse dynamics of a robot arm.

Since we are using stochastic optimization, the learning rates need to be chosen carefully. We found that the rates used in Hensman et al. (2013) also work well for our model. Specifically, we used the learning rates of 0.01 for the variational parameters, 1×10^{-5} for the covariance hyperparameters, and 1×10^{-4} for the weights, noise precisions, and inducing inputs. We also included a momentum term of 0.9 for all of the parameters except the variational parameters and the inducing inputs. All of the experiments are executed on an Intel(R) Core(TM) i7-2600 3.40GHz CPU with 8GB of RAM using Matlab R2012a.

4.1 TOY PROBLEM

In this toy problem, two related outputs are simulated from the same latent function $\sin(x)$ and corrupted by independent noise: $y_1(x) = \sin(x) + \epsilon$ and

$y_2(x) = -\sin(x) + \epsilon$, $\epsilon \sim \mathcal{N}(0, 0.01)$. Each output is given 200 observations with missing values in the $(-7, -3)$ interval for the first output and the $(4, 8)$ interval for the second output. We used $Q = 1$ latent sparse process with squared exponential kernel, $h_1(x) = h_2(x) = 0$, and $M = 15$ inducing inputs for our model.

Figure 1 shows the predictive distributions by our model (COGP) and independent GPs with stochastic variational inference (SVIGP, one for each output). The locations of the inducing inputs are fixed and identical for both methods. It is apparent from the figure that the independent GPs fail to predict the functions in the unobserved regions, especially for output 1. In contrast, by using information from the observed intervals of one output to interpolate the missing signal of the other, COGP makes perfect prediction for both outputs. This confirms the effectiveness of collaborative learning of sparse processes via the shared inducing variables. Additionally, we note that the inference procedure learned that the weights are $w_{11} = 1.07$ and $w_{21} = -1.06$ which accurately reflects the correlation between the two outputs.

4.2 FOREIGN EXCHANGE RATE PREDICTION

The first real world application we consider is to predict the foreign exchange rate w.r.t the US dollar of the top 10 international currencies (CAD, EUR, JPY, GBP, CHF, AUD, HKD, NZD, KRW, and MXN) and 3 precious metals (gold, silver, and platinum)¹. The setting of our experiment described here is identical to that in Álvarez et al. (2010). The dataset consists of all the data available for the 251 working days in the year of 2007. There are 9, 8, and 42 days of missing values for gold, silver, and platinum, respectively. We remove from the data the exchange rate of CAD on days 50–100, JPY on day 100–150, and AUD on day 150–200. Note that these 3 currencies are from very different geographical locations, making the problem more interesting. The 153 points are used for testing, and the remaining 3051 data points are used for training. Since the missing data corresponds to long contiguous sections, the objective here is to evaluate the capacity of the model to impute the missing currency values based on other currencies.

For preprocessing we normalized the outputs to have zero mean and unit variance. Since the exchange rates are driven by a small number of latent market forces (see e.g. Álvarez et al., 2010), we tried different values of $Q = 1, 2, 3$ and selected $Q = 2$ which gave the best model evidence (ELBO). We used the squared-

¹Data is available at <http://fx.sauder.ubc.ca>

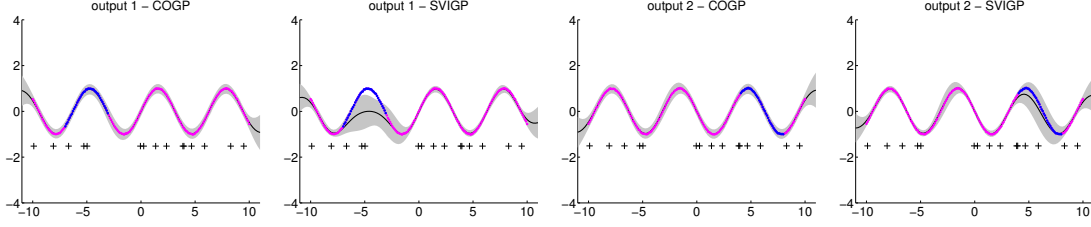


Figure 1: Simulated data and predictive distributions of by COGP (first and third figure) and independent GPs using stochastic variational inference (second and last figure) for the toy problem. Solid black line: predictive mean; grey bar: two standard deviations; magenta dots: real observations; blue dots: missing data. The black crosses show the locations of the inducing inputs. By sharing inducing points across the outputs, COGP accurately interpolates the missing function values.

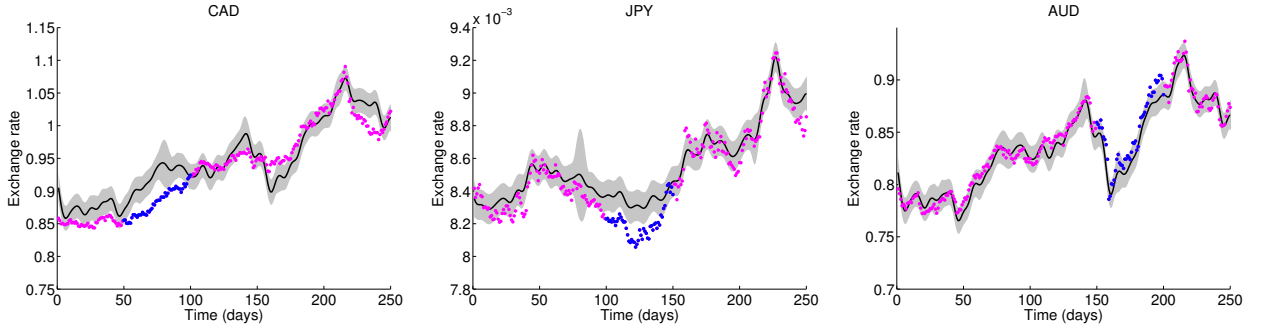


Figure 2: Real observations and predictive distributions for CAD (left), JPY (middle), and AUD (right). The model used information from other currencies to effectively extrapolate the exchange rates of AUD. The color coding scheme is the same as in Figure 1.

Table 2: Performance comparison on the foreign exchange rate dataset. Results are averages of the 3 outputs over 5 repetitions. Smaller figures are better.

METHOD	SMSE	NLPD
COGP	0.2125	-0.8394
CGP	0.2427	-2.9474
IGP	0.5996	0.4082

exponential covariance function for the shared processes and the noise covariance function for the individual process of each output. $M = 100$ inducing inputs (per sparse process) were randomly selected from the training data and fixed throughout training.

The real data and predictive distributions by our model are shown in Figure 2. They exhibit similar behaviors to those by the convolved model with inducing kernels in Álvarez et al. (2010). In particular, both models perform better at capturing the strong depreciation of the AUD than the fluctuations of the CAD and JPY currency. Further analysis of the dataset found that 4 other currencies (GBP, NZD, KRW, and MXN) also experienced the same trend during the days 150

– 200. This information from these currencies was effectively used by the model to extrapolate the values of the AUD.

We also report in Table 2 the predictive performance of our model compared to the convolved GPs model with exact inference (CGP, Alvarez and Lawrence, 2009) and independent GPs (IGP, one for each output). Our model outperforms both of CGP and IGP in terms of the standardized mean squared error (SMSE). CGP has lower negative log predictive density (NLPD), mainly due to the less conservative predictive variance of the exact CGP for the CAD currency. For reference, the convolved GPs with approximation via the variational inducing kernels (CGPVAR, Álvarez et al., 2010) has an SMSE of 0.2795 while the NLPD was not provided. Training took only 10 minutes for our model compared to 1.4 hours for the full CGP model.

4.3 AIR TEMPERATURE PREDICTION

Next we consider the task of predicting air temperature at 4 different locations in the south coast of England. The air temperatures are recorded by a network of weather sensors (named Bramblemet, Sotonmet, Cambermet, and Chimet) during the period from

Table 3: Performance comparison on the air temperature dataset. Results are averages of 2 outputs over 5 repetitions.

METHOD	SMSE	NLPD
COGP	0.1077	2.1712
CGP	0.1125	2.2219
IGP	0.8944	12.5319

July 10 to July 15, 2013. Measurements were taken every 5 minutes, resulting in a maximum of 4320 observations. There are missing data for Bramblemet (100 points), Chimet (15 points), and Sotonmet (1002 points), possibly due to network outages or hardware failures. We further simulated failure of the sensors by removing the observations from the time periods [10.2 - 10.8] for Cambermet and [13.5 - 14.2] for Chimet. The removed data comprises 375 data points and is used for testing. The remaining data consisting of 15,788 points is used for training. Similar to the previous experiment, the objective is to evaluate the ability of the model to use the signals from the functioning sensors to extrapolate the missing signals.

We normalized the outputs to have zero mean and unit variance. We used $Q = 2$ sparse processes with the squared exponential covariance function and individual processes with the noise covariance function. $M = 200$ inducing inputs were randomly selected from the training set and fixed throughout training.

The real data and the predictive distributions by our model, CGP with exact inference (Alvarez and Lawrence, 2009), and independent GPs are shown in Figure 3. It is clear that the independent GP model is clueless in the test regions and thus simply uses the average temperature as its prediction. For Cambermet, both COGP and CGP can capture the rising in temperature from the morning until the afternoon and the fall afterwards. The performance of the models are summarized in Table 3, which shows that our model outperforms CGP in terms of both SMSE and NLPD. It took 5 minutes on average to train our model compared to 3 hours of CGP with exact inference.

It is also worth noting the characteristics of the sparse processes learned by our model as they correspond to different patterns in the data. In particular, one process has an inverse lengthscale of 136 which captures the global increase in temperature during the training period while the other has an inverse lengthscale of 0.5 to model the local variations within a single day.

Table 4: Performance comparison on the robot inverse dynamics dataset. In the last two lines, standard GP is applied to output 1 and the other method is applied to output 2. Results are averaged over 5 repetitions.

METHOD	OUTPUT 1		OUTPUT 2	
	SMSE	NLPD	SMSE	NLPD
COGP, learn z	0.2631	3.0600	0.0127	0.8302
COGP, fix z	0.2821	3.2281	0.0131	0.8685
GP, SVIGP	0.3119	3.2198	0.0101	1.1914
GP, SOD	0.3119	3.2198	0.0104	1.9407

4.4 ROBOT INVERSE DYNAMICS

Our last experiment is with a dataset relating to an inverse dynamics model of a 7-degree-of-freedom anthropomorphic robot arm (Vijayakumar and Schaal, 2000). The data consists of 48,933 datapoints mapping from a 21-dimensional input space (7 joints positions, 7 joint velocities, 7 joint accelerations) to the corresponding 7 joint torques. It has been used in previous work (see e.g. Rasmussen and Williams, 2006; Vijayakumar and Schaal, 2000) but only for single task learning. Chai et al. (2008) considered multitask learning of robot inverse dynamics but on a different and much smaller dataset.

Here we consider joint learning for the 4th and 7th torques, where the former has 2,000 points while the latter has 44,484 points for training. The test set consists of 8,898 observations equally divided between the two outputs.

Since none of the existing multi-output models are applicable to problems of this scale, we compare with independent models that learn each output separately. Standard GP is applied to the first output as it has only 2,000 observations for training. For the second output, we used two baselines. The first is the subset of data (SOD) approach where 2,000 data points are randomly selected for training with a standard GP model. The second is the sparse GP with stochastic variational inference (SVIGP) using 500 inducing inputs and a batch size of 1,000. In case of COGP, we also used a batch size of 1,000 and 500 inducing points for the shared process ($Q = 1$) and each of the individual processes.

The performance of all methods in terms of SMSE and NLPD is given in Table 4. The benefits of learning the two outputs jointly are evident, as can be seen by the significantly lower SMSE and NLPD of COGP compared to the full GP for the first output (4th torque). While the SMSE of the second output is essentially the same for all methods, the NLPD of COGP is sub-

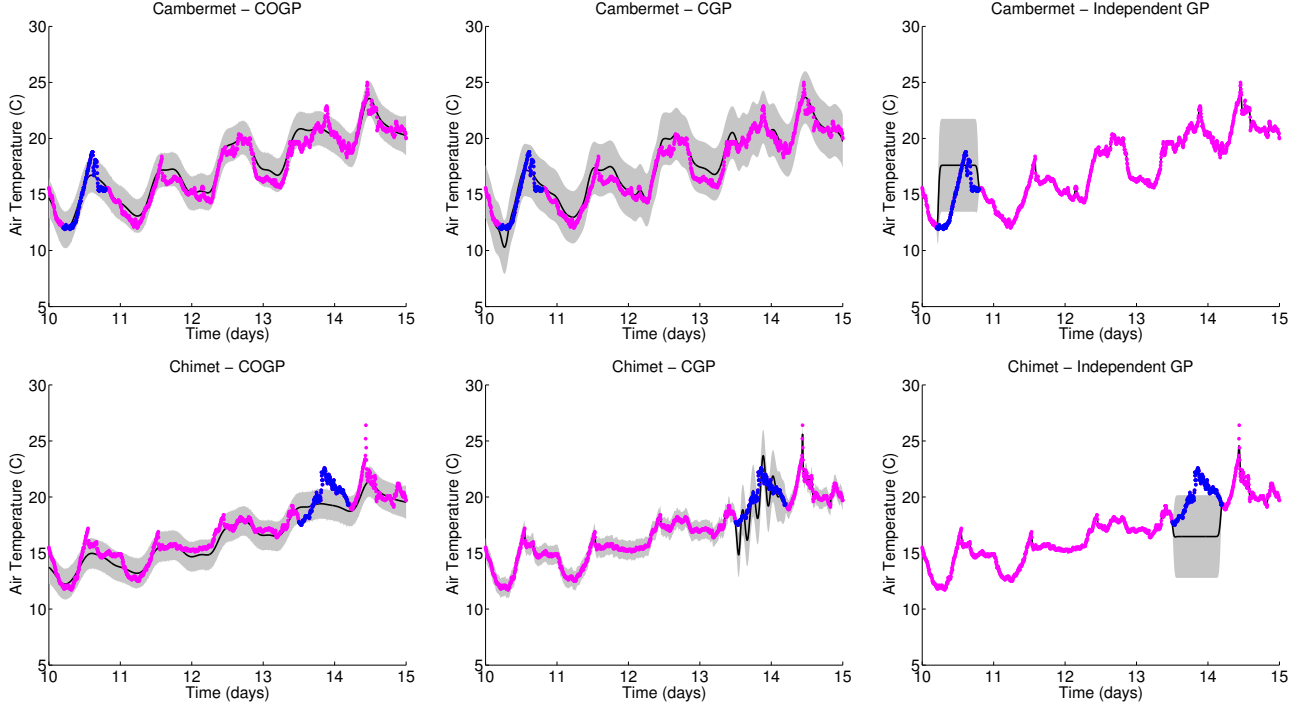


Figure 3: Real data and predictive distributions by our method (COGP, left figures), the convolved GP method with exact inference (CGP, middle figures), and full independent GPs (right figures) for the air temperature problem. The coding color scheme is the same as in Figure 1.

stantially better than that of the independent SVIGP model which has the same amount of training data for this torque. These results validate the impact of collaborative learning under sparsity assumptions, opening up new opportunities for improvement over single task learning with independent sparse processes.

Finally, we see on Table 4 that optimizing the inducing inputs can yield better performance than fixing them. More importantly, the overhead in computation is small, as demonstrated by the training times shown in Figure 4. For instance, the total training time is only 1.9 hours when learning with 500 inducing inputs compared to 1.6 hours when fixing them. As this dataset is 21-dimensional, this small difference in training time confirms that learning of the inducing inputs is a practical option even when dealing with problems of high dimensions.

5 DISCUSSION

We have presented scalable multi-output GPs for learning of correlated functions. The formulation around the inducing variables was shown to be conducive to effective and scalable joint learning under sparsity. We note that although our large scale experiments were done with over 40,000 observations – the largest publicly available multi-output dataset found,

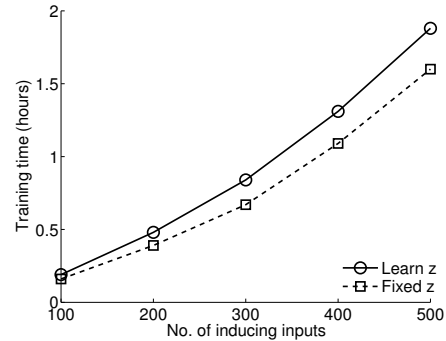


Figure 4: Learning of the inducing inputs is a practical option as the overhead in training time is small.

the model can easily handle much bigger datasets.

Acknowledgements

EVB thanks Stephen Hardy for early discussions on inducing-point sharing for multi-output GPs. NICTA is funded by the Australian Government through the Department of Communications and the Australian Research Council through the ICT Centre of Excellence Program.

References

- Alvarez, M. and Lawrence, N. D. (2009). Sparse convolved Gaussian processes for multi-output regression. In *NIPS*.
- Álvarez, M. A., Luengo, D., Titsias, M. K., and Lawrence, N. D. (2010). Efficient multioutput Gaussian processes through variational inducing kernels. In *AISTATS*.
- Bonilla, E. V., Chai, K. M. A., and Williams, C. K. I. (2008). Multi-task Gaussian process prediction. In *NIPS*.
- Boyle, P. and Frean, M. (2005). Dependent Gaussian processes. In *NIPS*.
- Chai, K. M. A., Williams, C. K., Klanke, S., and Vijayakumar, S. (2008). Multi-task gaussian process learning of robot inverse dynamics. In *NIPS*.
- Goovaerts, P. (1997). *Geostatistics for natural resources evaluation*. Oxford University Press.
- Hensman, J., Fusi, N., and Lawrence, N. D. (2013). Gaussian processes for big data. In *Uncertainty in Artificial Intelligence*.
- Hensman, J., Rattray, M., and Lawrence, N. D. (2012). Fast variational inference in the conjugate exponential family. In *NIPS*.
- Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine Learning*, 37:183–233.
- Nguyen, T. V. and Bonilla, E. V. (2013). Efficient variational inference for gaussian process regression networks. In *AISTATS*.
- Osborne, M. A., Roberts, S. J., Rogers, A., Ramchurn, S. D., and Jennings, N. R. (2008). Towards real-time information processing of sensor network data using computationally efficient multi-output Gaussian processes. In *Proceedings of the 7th international conference on Information processing in sensor networks*.
- Quiñonero-Candela, J. and Rasmussen, C. E. (2005). A unifying view of sparse approximate Gaussian process regression. *The Journal of Machine Learning Research*.
- Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. The MIT Press.
- Teh, Y. W., Seeger, M., and Jordan, M. I. (2005). Semiparametric latent factor models. In *AISTATS*.
- Titsias, M. (2009). Variational learning of inducing variables in sparse Gaussian processes. In *AISTATS*.
- Titsias, M. K. and Lázaro-Gredilla, M. (2011). Spike and slab variational inference for multi-task and multiple kernel learning. In *NIPS*.
- Vijayakumar, S. and Schaal, S. (2000). Locally weighted projection regression: An $O(n)$ algorithm for incremental real time learning in high dimensional space. In *ICML*.
- Wilson, A. G., Knowles, D. A., and Ghahramani, Z. (2012). Gaussian process regression networks. In *ICML*.

Combining predictions from linear models when training and test inputs differ

Thijs van Ommen

Centrum Wiskunde & Informatica
Amsterdam, The Netherlands

Abstract

Methods for combining predictions from different models in a supervised learning setting must somehow estimate/predict the quality of a model's predictions at unknown future inputs. Many of these methods (often implicitly) make the assumption that the test inputs are identical to the training inputs, which is seldom reasonable. By failing to take into account that prediction will generally be harder for test inputs that did not occur in the training set, this leads to the selection of too complex models. Based on a novel, unbiased expression for KL divergence, we propose XAIC and its special case FAIC as versions of AIC intended for prediction that use different degrees of knowledge of the test inputs. Both methods substantially differ from and may outperform all the known versions of AIC *even when the training and test inputs are iid*, and are especially useful for deterministic inputs and under covariate shift. Our experiments on linear models suggest that if the test and training inputs differ substantially, then XAIC and FAIC predictively outperform AIC, BIC and several other methods including Bayesian model averaging.

1 INTRODUCTION

In the statistical problem of model selection, we are given a set of models $\{\mathcal{M}_i \mid i \in \mathcal{I}\}$, each of the form $\mathcal{M}_i = \{g_i(\cdot \mid \theta) \mid \theta \in \Theta_i\}$, where the $g_i(\cdot \mid \theta)$ are density functions on (sequences of) data. We wish to use one of these models to explain our data and/or to make predictions of future data, but do not know which model explains the data best. It is well known that simply selecting the model containing the maximum likelihood distribution from among all the models leads to overfitting, so any expression of the quality of a model must somehow avoid this problem. One way to do this is by estimating each model's ability

to predict *unseen* data (this will be made precise below). This approach is used by many methods for model selection, including cross-validation, AIC (Akaike, 1973) and its many variants, Gelfand and Ghosh's D_k (1998), and BPIC (Ando, 2007). However, none of these methods takes into account that for supervised learning problems, the generalization error being estimated will vary with the test input variables. Instead, they implicitly assume that the test inputs will be *identical* to the training inputs.

In this paper, we derive an estimate of the generalization error that does take the input data into account, and use this to define a new model selection criterion XAIC, its special case FAIC, and the variants XAIC_C and FAIC_C (small sample corrections). We use similar assumptions as AIC, and thus our methods can be seen as relatives of AIC that are adapted to supervised learning when the training and test inputs differ. Our experiments show that our methods have excellent predictive performance, better even than Bayesian model averaging in some cases. Also, we show theoretically that AIC's unawareness of input variables leads to a bias in the selected model order, even in the seemingly safe case where the test inputs are drawn from the same distribution as the training inputs. No existing model selection method seems to address this issue adequately, making XAIC and FAIC more than "yet another version of AIC".

It is in fact quite surprising that, more than 40 years after its original invention, all the forms of AIC currently in use are biased in the above sense, and in theoretical analyses, conditional model selection methods are often even compared on a new point x constrained to be one of the x values in the training data (see e.g. Yang (2005)), even though in most practical problems, a new point x will *not* be drawn from this empirical training data distribution, but rather should be regarded as falling in one of the three cases considered in this paper: (a) it is drawn from the same distribution as the training data (but not necessarily equal to one of the training inputs); (b) it is drawn from a different distribution (covariate shift); (c) it is set to a fixed, observable value, usually not in the training set, but the process that gave rise

to this value may not be known.

1.1 GOALS OF MODEL SELECTION

When choosing among or combining predictions from different models, one can have different goals in mind. Whereas BIC and BMS (Bayesian model selection) focus on finding the most probable model, methods like AIC, cross-validation and SRM (structural risk minimization, Vapnik (1998)) aim to find the model that leads to the best *predictions* of future data. While AIC and cross-validation typically lead to predictions that converge faster to optimal in the sense of KL-divergence than those of BIC and BMS, it is also well-known that, unlike BIC and BMS, such methods are not statistically consistent (i.e. they do not find the smallest submodel containing the truth with probability 1 as $n \rightarrow \infty$); there is an inherent conflict between these two goals, see for example Yang (2007); Van Erven et al. (2007, 2012). Like AIC, the XAIC and FAIC methods developed here aim for predictive optimality rather than consistency, thus, if consistency is the main concern, they should not be used. We also stress at the outset that, unlike most other model selection criteria, the model selected by FAIC may *depend* on the new x whose corresponding y value is to be predicted; for different x , a different model may be selected based on the same training data. Since — as in many other model selection criteria — our goal is predictive accuracy rather than ‘finding the true model’, and since the dependence on the test x helps us to get substantially better predictions, we are not worried by this dependency.

FAIC thus cannot be said to select a ‘single’ model for a given training set — it merely outputs a *function* from x values to models. As such, it is more comparable with BMA (Bayesian model *averaging*) rather than BMS (*selection*). BMA is of course a highly popular method for data prediction; like FAIC, it adapts its predictions to the test input x (as we will see, FAIC tends to select a simpler model if there are not many training points near x ; BMA predicts with a larger variance if there are not many training points near x). BMA leads to the optimal predictions in the idealized setting where one takes expectation under the prior (i.e., in frequentist terms, we imagine nature to draw a model, and then a distribution within the chosen model, both from the prior used in BMA, and then data from the drawn distribution), and usually performs very well in practice as well. It is of considerable interest then that our XAIC and FAIC outperform Bayes by a fair margin in some of our experiments in Section 5.

1.2 IN-SAMPLE AND EXTRA-SAMPLE ERROR

Many methods for model selection work by computing some estimate of how well each model will do at predicting unseen data. This generalization error may be defined in various ways, and methods can further vary in the as-

sumptions used to find an estimate. AIC (Akaike, 1973) is based on the expression for the generalization error

$$-2 \mathbb{E}_{\mathbf{U}} \mathbb{E}_{\mathbf{V}} \log g_i(\mathbf{V} \mid \hat{\theta}_i(\mathbf{U})), \quad (1)$$

for model $\mathcal{M}_i = \{g_i(\cdot \mid \theta) \mid \theta \in \Theta_i\}$, where $\hat{\theta}_i(\mathbf{U})$ denotes the element of Θ_i which maximizes the likelihood of data \mathbf{U} , and where both random variables are independent samples of n data points each, both following the true distribution of the data. (We use capitals to denote sequences of data points, and boldface for random variables.) Throughout this paper, \log denotes the natural logarithm.) Up to an additive term which is the same for all models, the inner expectation is the KL divergence from the true distribution to $g_i(\cdot \mid \hat{\theta}_i(\mathbf{U}))$. An interpretation of (1) is that we first estimate the model’s parameters using a random sample \mathbf{U} , then judge the quality of this estimate by looking at its performance on an independent, identically distributed sample \mathbf{V} . AIC then works by estimating (1) for each model by the asymptotically unbiased estimator

$$-2 \log g_i(\mathbf{U} \mid \hat{\theta}(\mathbf{U})) + 2k, \quad (2)$$

and selecting the model minimizing this estimate. Thus AIC selects the model whose maximum likelihood estimate is expected to be closest to the truth in terms of KL divergence. In the sequel, we will consider only one model at a time, and therefore omit the model index.

In supervised learning problems such as regression and classification, the data points consist of two parts $u_i = (x_i, y_i)$, and the models are sets of distributions on the *output variable* y conditional on the *input variable* x (which may or may not be random). We call these *conditional* models. The conditionality expresses that we are not interested in explaining the behaviour of x , only that of y given x . Then (1) can be adapted in two ways: as the *extra-sample error*

$$-2 \mathbb{E}_{\mathbf{Y} \mid X} \mathbb{E}_{\mathbf{Y}' \mid X'} \log g(\mathbf{Y}' \mid X', \hat{\theta}(X, \mathbf{Y})), \quad (3)$$

and, replacing both X and X' by a single variable X , as the *in-sample error*

$$-2 \mathbb{E}_{\mathbf{Y} \mid X} \mathbb{E}_{\mathbf{Y}' \mid X} \log g(\mathbf{Y}' \mid X, \hat{\theta}(X, \mathbf{Y})), \quad (4)$$

where capital letters again denote sequences of data points. Contrary to (1), these quantities capture that the expected quality of a prediction regarding y may vary with x .

An example of a supervised learning setting is given by *linear models*. In a linear model, an input variable x is represented by a *design vector* and a sequence of n inputs by an $n \times p$ *design matrix*; with slight abuse of notation, we use x and X to represent these. Then the densities $g(\mathbf{Y} \mid X, \mu)$ in the model are Gaussian with mean $X\mu$ and covariance matrix $\sigma^2 I_n$ for some fixed σ^2 . Because g is of the form $e^{-\text{squared error}}$, taking the negative logarithm as in (1) produces an expression whose main component is a sum of

squared errors; the residual sum of squared errors $\text{RSS}(\mathbf{Y})$ is the minimum for given data, which is attained by the maximum likelihood estimator. Alternatively, σ^2 may be another parameter in addition to μ if the true variance is unknown.

It is standard to apply ordinary AIC to supervised learning problems, for example for linear models with fixed variance where (2) takes the well-known form

$$\frac{1}{\sigma^2} \text{RSS}(\mathbf{Y}) + 2k, \quad (5)$$

where k is the number of parameters in the model. But because the standard expression behind AIC (1) makes no mention of X or X' , this corresponds to the tacit assumption that $X = X'$, so that the in-sample error is being estimated.

However, the extra-sample error is more appropriate as a measure of the expected performance on new data. AIC was intended to correct the bias that results from evaluating an estimator on the data from which it was derived, but because it uses the in-sample error, AIC evaluates estimators on new output data, but old input data. So we see that in supervised problems, a bias similar to the one it was intended to correct is still present in AIC.

1.3 CONTENTS

The remainder of this article is structured as follows. In Section 2, we develop our main results about the extra-sample error and propose a new model selection criterion based on this. It involves $\kappa_{X'}$, a term which can be calculated explicitly for linear models; we concentrate on these models in the remainder of the paper. Special cases of our criterion, including a focused variant, are presented in Section 3. In Section 4 we discuss the behaviour of our estimate of the extra-sample error, and find that without our modification, AIC's selected model orders are biased. Several experiments on simulated data are described in Section 5. Section 6 contains some further theoretical discussion regarding Bayesian prediction and covariate shift. Finally, Section 7 concludes. All proofs are in the supplementary material.¹

2 ESTIMATING THE EXTRA-SAMPLE ERROR

In this section, we will derive an estimate for the extra-sample error. Our assumptions will be similar to those used in AIC to estimate the in-sample error; therefore, we start with some preliminaries about the setting of AIC.

2.1 PRELIMINARIES

In the setting of AIC, the data points are independent but not necessarily identically distributed. The number of data points in \mathbf{Y} and \mathbf{Y}' is n . We define the Fisher information matrix $I(\theta)$ as $-\mathbb{E}_{\mathbf{Y}'} \frac{\partial^2}{\partial \theta^2} \log g(\mathbf{Y}' | \theta)$, and define the conditional Fisher information matrix $I(\theta | X')$ analogously. We write $\text{Cov}(\hat{\theta}(X, \mathbf{Y}) | X)$ for the conditional covariance matrix $\mathbb{E}_{\mathbf{Y}|X}[\hat{\theta}(X, \mathbf{Y}) - \mathbb{E}_{\mathbf{Y}|X} \hat{\theta}(X, \mathbf{Y})][\hat{\theta}(X, \mathbf{Y}) - \mathbb{E}_{\mathbf{Y}|X} \hat{\theta}(X, \mathbf{Y})]^\top$.

Under standard regularity assumptions, there exists a unique parameter value θ_o that minimizes the KL divergence from the true distribution, and this is what $\hat{\theta}(\mathbf{Y})$ converges to. Under this and other (not very restrictive) regularity assumptions (Shibata, 1989), it can be shown that (Burnham and Anderson, 2002)

$$-2 \log g(\mathbf{Y} | \hat{\theta}(\mathbf{Y})) + 2 \widehat{\text{tr}} \left\{ I(\theta_o) \text{Cov}(\hat{\theta}(\mathbf{Y})) \right\} \quad (6)$$

(where $\widehat{\text{tr}}$ represents an appropriate estimator of that trace) is an asymptotically unbiased estimator of (1). The model selection criterion TIC (Takeuchi's information criterion) selects the model which minimizes (6).

The estimator of the trace term that TIC requires has a large variance, making it somewhat unreliable in practice. AIC uses the very simple estimate $2k$ for TIC's trace term. This estimate is generally biased except when the true data-generating distribution is in the model, but obviously has 0 variance. Also, if some models are more misspecified than others, those models will have a worse log-likelihood. This term in AIC grows linearly in the sample size, so that asymptotically, those models will be disqualified by AIC. Thus AIC selects good models even when its penalty term is biased due to misspecification of the models.

This approach corresponds to making the following assumption in the derivation leading to AIC's penalty term:

Assumption 1 *The model contains the true data-generating distribution.*

It follows that θ_o specifies this distribution. We emphasize that this assumption is only required for AIC's derivation and does not mean that AIC necessarily works badly if applied to misspecified models. Under this assumption, the two matrices in (6) cancel, so the objective function becomes (2), the standard formula for AIC (Burnham and Anderson, 2002).

We now move to supervised learning problems, where the true distribution of the data and the distributions g in the models are conditional distributions of output values given input values. In this setting, the data are essentially iid in the sense that $g(\mathbf{Y} | X, \theta) = \prod_{i=1}^n g(\mathbf{y}_i | x_i, \theta)$. That is, the outputs are independent given the inputs, and if two input variables are equal, the corresponding output variables

¹Posted on arXiv

are identically distributed. Also, the definition of θ_o would need to be modified to depend on the training inputs, but since Assumption 1 now implies that $g(\mathbf{y} \mid x, \theta_o)$ defines the true distribution of \mathbf{y} given x for all x , we can take this as the definition of θ_o for supervised learning when Assumption 1 holds.

For supervised learning problems, AIC and TIC silently assume that X' either equals X or will be drawn from its empirical distribution. We want to remove this assumption.

2.2 MAIN RESULTS

We will need another assumption:

Assumption 2 For training data (X, \mathbf{Y}) and (unobserved) test data (X', \mathbf{Y}') ,

$$\begin{aligned} -\frac{1}{n} \mathbb{E}_{\mathbf{Y}|X} \log g(\mathbf{Y} \mid X, \theta_o) \\ = -\frac{1}{n'} \mathbb{E}_{\mathbf{Y}'|X'} \log g(\mathbf{Y}' \mid X', \theta_o), \end{aligned}$$

where n and n' denote the number of data points in X and X' , respectively.

This assumption ensures that the log-likelihood on the test data can be estimated from the training data. If \mathbf{X} and \mathbf{X}' are random and mutually iid, this is automatically satisfied when the expectations are taken over these inputs as well. While this assumption of randomness is standard in machine learning, there are other situations where X and X' are not random and Assumption 2 holds nevertheless. For instance, this is the case if $g(\mathbf{y} \mid x, \theta)$ is such that $\mathbf{y}_i = f_\theta(x_i) + \mathbf{z}_i$, where the noise terms \mathbf{z}_i are zero-mean and iid (their distribution may depend on θ). This additive noise assumption is common in regression-like settings. Then Assumption 1 implies that Assumption 2 holds for all X, X' .

To get an estimator of the extra-sample error (3), we do not make any assumptions about the process generating X and X' but leave the variables free. We allow $n \neq n'$.

Theorem 1 Under Assumptions 1 and 2 and some standard regularity conditions (detailed in the supplementary material), and for n' either constant or growing with n ,

$$\begin{aligned} -2 \frac{n}{n'} \mathbb{E}_{\mathbf{Y}|X} \mathbb{E}_{\mathbf{Y}'|X'} \log g(\mathbf{Y}' \mid X', \hat{\theta}(X, \mathbf{Y})) \\ = -2 \mathbb{E}_{\mathbf{Y}|X} \log g(\mathbf{Y} \mid X, \hat{\theta}(X, \mathbf{Y})) + k + \kappa_{X'} + o(1), \end{aligned} \quad (7)$$

where $\kappa_{X'} = \frac{n}{n'} \text{tr} \left\{ I(\theta_o \mid X') \text{Cov}(\hat{\theta}(X, \mathbf{Y}) \mid X) \right\}$.

Moreover, if the true conditional distribution of \mathbf{Y} given X is Gaussian with fixed variance and the conditional distributions in the models are also Gaussian with that same

variance (as is the case in linear models with known variance), then the above approximation becomes exact.

We wish to use (7) as a basis for model selection. To do this, first note that (7) can be estimated from our training data using

$$-2 \log g(\mathbf{Y} \mid X, \hat{\theta}(X, \mathbf{Y})) + k + \kappa_{X'}. \quad (8)$$

Theorem 1 expresses that this is an asymptotically unbiased estimator of the extra-sample error. We see that the difference with standard AIC (2) is that the penalty $2k$ has been replaced by $k + \kappa_{X'}$. We propose to use (8) as the basis for a new model selection criterion *extra-sample AIC (XAIC)*, which chooses the model that minimizes an estimator of (8). What remains for this is to evaluate $\kappa_{X'}$, which may depend on the unknown true distribution, and on the test set through X' .

2.3 THE $\kappa_{X'}$ AND $o(1)$ TERMS FOR LINEAR MODELS

If the densities g are Gaussian, then $\kappa_{X'}$ does not depend on the unknown θ_o because the Fisher information is constant, so no additional estimation is necessary to evaluate it. Thus for a linear model with fixed variance, $\kappa_{X'}$ becomes

$$\begin{aligned} \kappa_{X'} &= \frac{n}{n'} \text{tr} \left\{ \left[\frac{1}{\sigma^2} X'^\top X' \right] \left[\sigma^2 (X^\top X)^{-1} \right] \right\} \\ &= \frac{n}{n'} \text{tr} \left[X'^\top X' (X^\top X)^{-1} \right]. \end{aligned}$$

If the variance is also to be estimated, it can be easily seen that $\kappa_{X'}$ will become this value plus one. In that case, the approximation in Theorem 1 is not exact (as it is in the known variance case), but the $o(1)$ term can be evaluated explicitly:

Theorem 2 For a linear model with unknown variance,

$$\begin{aligned} -2 \frac{n}{n'} \mathbb{E}_{\mathbf{Y}|X} \mathbb{E}_{\mathbf{Y}'|X'} \log g(\mathbf{Y}' \mid X', \hat{\theta}(X, \mathbf{Y})) \\ = -2 \mathbb{E}_{\mathbf{Y}|X} \log g(\mathbf{Y} \mid X, \hat{\theta}(X, \mathbf{Y})) \\ + k + \kappa_{X'} + \frac{(k + \kappa_{X'})(k + 1)}{n - k - 1}, \end{aligned}$$

where $\kappa_{X'}$ can again be computed from the data and equals $(n/n') \text{tr}(X'^\top X' (X^\top X)^{-1}) + 1$, and k is the number of parameters including σ^2 .

Theorem 2 presents an extra-sample analogue of the well-known small sample correction AIC_C (Hurvich and Tsai, 1989), which is derived similarly and uses a penalty of $2k + 2k(k + 1)/(n - k - 1)$. We define XAIC_C accordingly. Though the theorem holds exactly only in the specific case described, we believe that the extra penalty term will lead to better results in much more general settings in practice, as is the case with AIC_C (Burnham and Anderson, 2002).

3 MODEL SELECTION FOR EXTRA-SAMPLE PREDICTION

In this section, we discuss several concrete model selection methods, all based on the XAIC formula (8) and thus correcting AIC's bias.

3.1 NONFOCUSED VERSIONS OF XAIC

Except in trivial cases, the extra-sample error (3) and its estimate (8) depend on the test inputs X' , so some knowledge of X' is required when choosing a model appropriate for extra-sample prediction. In a semi-supervised learning setting where X' itself is known at the time of model selection, we could evaluate (8) directly for each model. However, X' might not yet be known when choosing a model.

If \mathbf{X}' is not known but its distribution is, we can replace $\kappa_{\mathbf{X}'}$ by its expectation; for iid inputs, computing this reduces to computing $\mathbb{E}_{\mathbf{X}'} I(\theta_o | \mathbf{x}')$.

If the distribution of \mathbf{X}' is also unknown, we need to estimate it somehow. If it is believed that \mathbf{X} and \mathbf{X}' follow the same distribution, the empirical distribution of \mathbf{X} could be used as an estimate of the distribution of \mathbf{X}' . Then AIC is retrieved as a special case. Section 4 will show that this is a bad choice even if \mathbf{X} and \mathbf{X}' follow the same distribution, so a smoothed estimate is recommended instead.

Of course, we are not restricted to the case where \mathbf{X} and \mathbf{X}' follow similar distributions. In the setting of covariate shift (Sugiyama and Kawanabe, 2012), the distributions are different but known (or can be estimated). This variant of XAIC is directly applicable to that setting, yielding an unbiased analogue of AIC.

3.2 FOCUSED MODEL SELECTION

It turns out there is a way to apply (8) even when nothing is known about the process generating X and X' . If our goal is prediction, we can set X' to the single point x' for which we need to predict the corresponding y' . Contrary to standard model selection approaches, we thus use x' already at the stage of model selection, rather than only inside the models. We define the model selection criterion *Focused AIC* (FAIC) as this special case of XAIC, and FAIC_C as its small sample correction.

A focused model selection method implements the intuition that those test points whose input is further away from the training inputs should be predicted with more caution; that is, with less complex models. As discussed in Section 1.1, methods that optimize predictive performance often are not consistent; this hurts in particular for test inputs far away from the training inputs. We expect that extra-sample adaptations of such methods (like XAIC) are also inconsistent, but that using the focused special case helps

to guard against this small chance of large loss.

Choosing a model specifically for the task at hand potentially lets us end up with a model that performs this task much better than a model found by a non-focused model selection method. However, there are situations in which focus is not a desirable property: the mapping from input values to predictions given by a focused model selection method will be harder to interpret than that of a non-focused method, as it is a combination of the models under consideration rather than a single one of them. Thus, if the experimenter's goal is interpretation/transparency, a focused model selection method is not recommended; these methods are best applied when the goal is prediction.

Evaluating the x' -dependent model selection criterion separately for each x' leads to a regression curve which in general will not be from any one of the candidate models, but only piecewise so. It will usually have discontinuities where it switches between models. If the models contain only continuous functions and such discontinuities are undesirable, Akaike weights (Akaike, 1979; Burnham and Anderson, 2002) may be used to get a continuous analogue of the FAIC regression curve.

4 AIC VS XAIC (k VS κ_x) IN LINEAR MODELS

Intuitively, the quantity κ_x that appears as a penalty term in the XAIC formula (8) expresses a measure of dissimilarity between the test input x and the training inputs X . This measure is determined fully by the models and does not have to be chosen some other way. However, its properties are not readily apparent. Because κ_x can be computed exactly for linear models, we investigate some of its properties in that case.

One useful characterization of κ_x is the following: if we express the design vector x of the test point in a basis that is orthonormal to the empirical measure of the training set X , then $\kappa_x = \|x\|^2$.

For given X , x may exist such that κ_x is either greater or smaller than the number of parameters k . An example of $\kappa_x < k$ occurs for the linear model consisting of all linear functions with known variance (so $k = 2$). Then κ_x will be minimized when x lies at the mean of the input values in the training set, where $\kappa_x = 1$.

We will now consider the case where \mathbf{X} and \mathbf{x} are random and iid. We showed that the XAIC expression (8) is an unbiased estimator of the extra-sample error. AIC uses k in place of $\kappa_{\mathbf{x}}$, and the above suggests the possibility that maybe the instances where $\kappa_{\mathbf{x}} > k$ and those where $\kappa_{\mathbf{x}} < k$ cancel each other out, so that AIC would also be approximately unbiased as an estimate of the extra-sample error. However, the following proposition shows

that, except in a trivial case, $\kappa_{\mathbf{x}}$ is on average greater than k . This means that in those settings, AIC underestimates the model’s extra-sample error.

(We should mention here that if \mathbf{X} and \mathbf{x} are random and mutually iid, then as $n \rightarrow \infty$, AIC’s bias goes to 0. The bias we show below concerns all finite n ; additionally, without focus, an extreme \mathbf{x} could result in a very biased AIC value even for large n .)

Proposition 3 *Consider a linear model \mathcal{M} with training inputs \mathbf{X} and test input \mathbf{x} iid such that $\mathbf{X}^\top \mathbf{X}$ is almost surely invertible. Let \mathcal{M}' be the submodel obtained by removing the final entry from every design vector. Then these models are related by $\mathbb{E} \kappa_{\mathbf{x}} \geq \mathbb{E} \kappa_{\mathbf{x}'} + 1$, with strict inequality if \mathbf{x} has at least two entries.*

It follows by induction on k that for random input data, AIC is biased as an estimate of the extra-sample error except in a special case with $k = 1$. Also, the bias becomes worse for larger models. This last fact is distressing, as it shows that when AIC assesses a sequence of nested models, the amount by which it overestimates their generalization ability grows with the model order. Thus the biases in the AIC scores lead to a bias in the selected model order, which was not evident from earlier work.

The XAIC formula (8) contains two terms that depend on the data: minus two times the log-likelihood, and the penalty term $\kappa_{X'}$. The log-likelihood measures distances between output values and is independent of X' , while $\kappa_{X'}$ expresses a property of input values and is largely unaffected by output values; in fact, in linear models its computation does not involve any (estimates based on) output values. Hence the variance of XAIC is no greater than that of AIC when comparing the two on fixed X, X' , so that XAIC’s reduction in bias does not come at the price of an increase in variance. However, focused model selection demands that X' is *not* held fixed, so that FAIC may have a larger variance than AIC. Similarly, if the distribution of \mathbf{X}' is being estimated as part of applying XAIC, the used estimator’s quality will affect the accuracy of the estimated generalization error.

5 EXPERIMENTS

We will now experimentally compare XAIC and FAIC (or more precisely, their small-sample corrected versions XAIC_C and FAIC_C) to several other model selection methods, in univariate and multivariate problems.

5.1 DESCRIPTION OF EXPERIMENTS

In the univariate experiments, linear models $\mathcal{M}_1, \dots, \mathcal{M}_7$ with unknown variance were considered. Model \mathcal{M}_i contained polynomials of degree $i - 1$ (and so had $i + 1$ parameters). The input values x of the training data were drawn

from a Gaussian distribution with mean 0 and variance 1, while the output values were generated as $\mathbf{y}_i = f(x_i) + \mathbf{z}$ with \mathbf{z}_i iid Gaussians with mean 0 and variance 0.1, and f some unknown true function. Given 100 training data points, each of the eight model selection methods under consideration had to select a model. The squared risk $(\hat{y} - f(x))^2$ of the chosen model’s prediction \hat{y} was computed for each of a range of values of the test point’s x , averaged over 100 draws of the training data. This experiment was performed for two different true functions: $f_1(x) = x + 2$ and $f_2(x) = |x|$.

In the multivariate experiments, each input variable was a vector (u_1, \dots, u_6) , and the models corresponded to all possible subsets of these 6 variables. Each model also included an intercept and a variance parameter. The true function was given by $f(u) = 2 + u_1 + 0.1u_2 + 0.03u_3 + 0.001u_4 + 0.003u_5$, and the additive noise was again Gaussian with variance 0.1. A set of $n' = 400$ test inputs was drawn from a standard Gaussian distribution, but the training inputs were generated differently in each experiment: from the same Gaussian distribution as the test inputs; from a uniform distribution on $[-\sqrt{3}, \sqrt{3}]^6$; or from a uniform ‘spike-and-slab’ mixture of two Gaussians with covariance matrices $(1/5)I_6$ and $(9/5)I_6$. Note that all three distributions have the same mean and covariance as the test input distribution, making these mild cases of covariate shift. For the Gaussian training case, we report the results for $n = 60$ and, after extending the same training set, for $n = 100$. Squared risks were averaged over the test set and further over 50 repeats of these experiments.

The experiments used the version of XAIC that is given a distribution of the test inputs, but not the test inputs themselves. In the multivariate experiments, XAIC used the actual (Gaussian) distribution of the test inputs. In the univariate case, two instances of XAIC were evaluated: one for test inputs drawn from the same distribution as the training inputs (standard Gaussian), and another (labelled XAIC_{C2}) for a Gaussian test input distribution with mean 0 and variance 4.

Bayesian model averaging (BMA) differs from the other methods in that it does not select a single model, but formulates its prediction as a weighted average over them; in our case, its prediction corresponds to the posterior mean over all models. Weighted versions exist of other model selection methods as well, such as Akaike weights (Akaike, 1979; Burnham and Anderson, 2002) for AIC and variants. In our experiments we saw that these usually perform similar to but somewhat better than their originals. In our univariate experiments, we decided against reporting these, as they are less standard. However, in the multivariate experiments, the weighted versions were all better than their selection counterparts, so both are reported separately to allow fair comparisons.

In our experiments, BMA used a uniform prior over the models. Within the models, Jeffreys' noninformative prior (for which the selected μ would correspond to the maximum likelihood $\hat{\mu}$ used by other methods) was used for the variable selection experiments; for the polynomial case, it proved too numerically unstable for the larger models, so there BMA uses a weakly informative Gaussian prior (variance 10^2 on μ_2, \dots, μ_7 with respect to the Hermite polynomial basis, and Jeffreys' prior on σ^2).

Of the model selection methods included in our experiments, AIC was extensively discussed in Section 2.1; as with XAIC and FAIC, we use here the small sample correction AIC_C (see Section 2.3). BIC (Schwarz, 1978) and BMS were mentioned in Section 1.1 as methods that attempt to find the most probable model given the data rather than aiming to optimize predictive performance; both are based on BMA, which computes the Bayesian posterior probability of each model. Three other methods were evaluated in our experiments; these are discussed below.

Like AIC, the much more recent focused information criterion (FIC) (Claeskens and Hjort, 2003) is designed to make good predictions. Unlike other methods, these predictions are for a *focus parameter* which may be any function of the model's estimate, not just its prediction at some input value (though we only used the latter in our experiments). Unlike FAIC, it uses this focus not just for estimating a model's variance, but also its bias; FAIC on the other hand uses a global estimate of a model's bias based on Assumption 2. A model's bias for the focus parameter is evaluated by comparing its estimate to that of the most complex model available.

Another more recent method for model selection is the subspace information criterion (SIC) (Sugiyama and Ogawa, 2001), which is applicable to supervised learning problems when our models are subspaces of some Hilbert space of functions, and our objective is to minimize the squared norm. Like FIC, SIC estimates the models' biases by comparing their estimates to that of a larger model, but it includes a term to correct for this large model's variance. In our experiments, we used the corrected SIC (cSIC) which truncates the bias estimate at 0.

Generalized cross-validation (GCV) (Golub et al., 1979) can be seen as a computationally efficient approximation of leave-one-out cross-validation for linear models. We included it in our experiments because Leeb (2008) shows that it performs better than other model selection methods when the test input variables are newly sampled.

5.2 RESULTS

Results from the two univariate experiments are shown in Figures 1 and 2 (squared risks) and in Table 1 (selected models). Squared risk results for the multivariate experi-

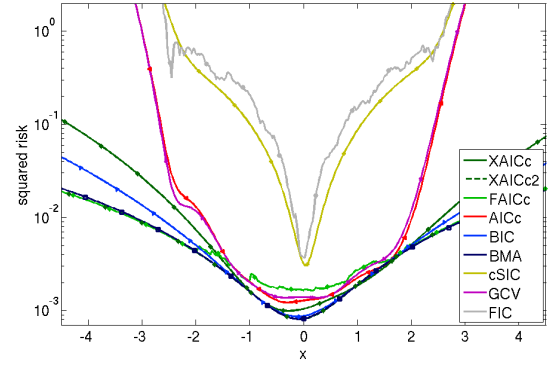


Figure 1: Squared risk of different model selection methods as a function of x when the true function is $f_1(x) = x + 2$.

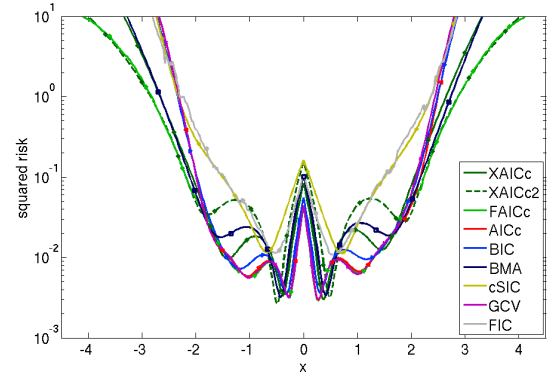


Figure 2: Squared risk of different model selection methods as a function of x when the true function is $f_2(x) = |x|$.

ments are given in Table 2 for the model selection methods, and in Table 3 for the model weighting/averaging variants.

XAIC and FAIC The characteristic behaviours of our methods are clearly visible in the univariate experiments. Both instances of XAIC perform well overall in both experiment. Of the two, XAIC_{C2} was set up to expect test inputs further away from the center. As a result, it selects models more conservatively, and obtains smaller risk at such off-center test inputs. Its selections were very stable: in both experiments, XAIC_{C2} selected the same model in each of the 100 runs.

We see that in the center of Figure 2, the simple model chosen by XAIC_{C2} was outperformed by more complex models. FAIC exploits this by choosing a model adaptively for each test input. This resulted in good risk performance at all test inputs.

In the multivariate experiments, FAIC was the best method for the spike-and-slab training data, where there are pronounced differences in training point density surrounding different test points, so that selecting a different model for

Table 1: Average selected model index per method for f_1 and f_2 , at test inputs $x' = 0$ and 4 (if different).

	XAIC _C	XAIC _{C2}	AIC _C	BIC	BMS	cSIC	GCV
f_1	2.10	2.00	2.33	2.02	2.00	2.94	2.38
f_2	4.57	3.00	6.38	5.70	4.05	4.70	6.49

	FAIC _C		FIC	
	$x' = 0$	$x' = 4$	$x' = 0$	$x' = 4$
f_1	2.94	2.00	2.66	3.12
f_2	6.56	1.54	5.29	5.35

Table 2: Multivariate: squared risk for different training sets; model selection

	Gaussian ($n = 60$)	uniform ($n = 60$)	spike- and-slab ($n = 60$)	Gaussian ($n = 100$)
XAIC _C	0.0119	0.0123	0.0144	0.0070
FAIC _C	0.0123	0.0127	0.0133	0.0077
AIC _C	0.0125	0.0126	0.0156	0.0070
BIC	0.0113	0.0128	0.0140	0.0073
BMS	0.0120	0.0126	0.0138	0.0075
cSIC	0.0119	0.0134	0.0138	0.0074
GCV	0.0129	0.0131	0.0153	0.0072
FIC	0.0196	0.0189	0.0241	0.0111

each pays off. The performance of XAIC was more reliable overall, comparing very favourably to each of its competitors.

AIC Our methods XAIC and FAIC were derived as adaptations of AIC, and share its tendency to go for complex models as soon as there is some indication that their predictions might be worthwhile. This leads to good predictions on average, but also to inconsistency: when a simpler model contains the true distribution, AIC will continue to select more complex models with positive probability, no matter how large n grows. This may sometimes hurt predictive performance, because the accuracy of the estimated parameter will be smaller for more complex models; for details, we refer to Yang (2007); Van Erven et al. (2007, 2012). XAIC makes a better assessment of the generalization error, even when the training and test inputs follow the same distribution, so that it overfits less than AIC and may achieve much better risks. FAIC differs from AIC in another way: its tendency to choose more complex models is strengthened in areas where many data points are available (so that the potential damage of picking an overly complex model is smaller), while it is suppressed when few data points are available (and the potential damage is much greater).

This tendency is also apparent in Table 1. In the first experiment, where a small model contains the true distribution,

Table 3: Multivariate: squared risk for different training sets; model weighting/averaging

	Gaussian ($n = 60$)	uniform ($n = 60$)	spike- and-slab ($n = 60$)	Gaussian ($n = 100$)
XAIC _{Cw}	0.0099	0.0108	0.0114	0.0063
FAIC _{Cw}	0.0100	0.0110	0.0110	0.0066
AIC _{Cw}	0.0101	0.0108	0.0119	0.0063
BICw	0.0096	0.0106	0.0111	0.0062
BMA	0.0100	0.0107	0.0113	0.0061

it causes FAIC to perform worse than AIC near $x = 0$. However, note that the vertical axis is logarithmic, so the difference appears larger than it is: when we average over the training input distribution, we find that FAIC performs better by a factor 20 in terms of squared risk.

In the multivariate experiments, XAIC again performs better than AIC, though the difference eventually disappears as n grows. With the notable exception of the spike-and-slab experiment, FAIC does not perform well here: in two of the experiments, it does worse than AIC. Part of the reason must be our observation at the end of Section 4: FAIC’s estimate of the generalization error, while unbiased, may potentially have a larger variance than (X)AIC’s estimate, and this is not always a good trade-off.

BIC and BMS/BMA BIC and BMS do not try to identify the model that will give the best predictions now, but instead attempt to find the most probable model given the data, which usually amounts to the simplest model containing the true distribution. This leads them to be conservative about selecting complex models. For similar reasons, Bayesian model averaging (BMA) puts only small weight on complex models. We see this in Figure 1, where BIC and BMA have good performance because they most often select the optimal second model (or in the case of BMA, give it the largest weight). However, for f_2 in Figure 2, it may be outperformed by FAIC or XAIC for test inputs away from the center. In the multivariate experiments, XAIC often performs better than BMS/BMA, and rarely much worse; the only instance of the latter is for the spike-and-slab data, where FAIC outperforms both. (See Section 6.1 for further discussion of BMA.)

FIC In all our experiments, FIC obtained large squared risks, and we see in Table 1 that its selection behaviour was the opposite of FAIC: for extreme x , FIC often selects a more complex model than near $x = 0$. This seems to happen because FIC uses the most complex model’s prediction at a given x to estimate each other model’s bias. Because the most complex model will usually have a significant variance, this resulted in FIC being misled in many of the experiments we examined. In particular, in areas with few

training inputs, FIC apparently usually believes the simpler models will perform badly because it attributes to them a large bias, so that the same model as elsewhere (or even a more complex one) is selected. Conversely, FIC was often observed to switch to an overly simple model near some input value where this model’s estimate happened to coincide with that of the most complex model.

SIC SIC obtained large risks in the univariate experiments due to underfitting. Its results in three of the four multivariate experiments were competitive, however.

GCV Based on Leeb (2008), we expected GCV might be one of the strongest competitors to XAIC. This was not clearly reflected in our experiments, where its performance was very similar to that of AIC.

6 DISCUSSION

6.1 RELATION TO THE BAYESIAN PREDICTIVE DISTRIBUTION

The quantity $\kappa_{x'}$ that occurs in FAIC has an interpretation in the Bayesian framework. If we do linear regression with known variance and a noninformative prior on μ , then after observing X , Y and x' , the predictive distribution of y' is $y' \mid Y, X, x' \sim \mathcal{N}(x'^\top \hat{\mu}, \sigma^2(1 + x'^\top (X^\top X)^{-1} x'))$. We see that $\kappa_{x'}$ and the variance of this predictive distribution obey a linear relation. Thus if BMA is allowed to give a distribution over output values as its prediction, then this distribution (a mixture of Gaussians with different variances) will reflect that some models’ predictions are more reliable than others. However, if the predictive distribution must be summarized by a point prediction, then such information is likely to be lost. For instance, if the point prediction \hat{y}' is to be evaluated with squared loss and \hat{y}' is chosen to minimize the expected loss under the predictive distribution (as in our experiments in Section 5), then \hat{y}' is a weighted average of posterior means for y' given x' (one mean for each model, weighted by its posterior probability). The predictive variances are not factored into \hat{y}' , so that in this scenario, BMA does not use the information captured by $\kappa_{X'}$ that XAIC and FAIC rely on.

This is not to say that BMA *should* use this information: the consideration of finding the most probable model (BMS, BIC) or the full distribution over models (BMA) is not affected by the purpose for which the model will be used, so it should not depend on the input values in the test data through $\kappa_{X'}$. This suggests that there is no XBIC analogue to XAIC. For Bayesian methods such as DIC (Spiegelhalter et al., 2002) and BPIC (Ando, 2007) that aim for good predictions, on the other hand, extra-sample and focused equivalents may exist.

6.2 RELATION TO COVARIATE SHIFT

We observed at the end of Section 4 that of the two data-dependent terms in XAIC, the log-likelihood is independent of X' , while $\kappa_{X'}$ is (largely) unaffected by output values. An important practical consequence of this split between input and output values is that XAIC and FAIC look for models that give a good overall fit, not just a good fit at the test inputs. X' is then used to determine how well we can expect these models to generalize to the test set. So if we have two models and believe each to be able to give a good fit in a different region of the input space, then FAIC is not the proper tool for the task of finding these regions: FAIC considers global fit rather than local fit when evaluating a model, and within the model selects the maximum likelihood estimator, not an estimator specifically chosen for a local fit at input point x .

In this respect, our methods differ from those commonly used in the covariate shift literature (see Sugiyama and Kawanabe (2012); Pan and Yang (2010); some negative results are in Ben-David et al. (2010)), where typically a model (and an estimator within that model) is sought that will perform well on the test set only, using for example importance weighting. This is appropriate if we believe that no available model can give satisfactory results on both training and test inputs simultaneously. In situations where such models are believed to exist, our methods try to find them using all information in the training set.

7 CONCLUSIONS AND FUTURE WORK

We have shown a bias in AIC when it is applied to supervised learning problems, and proposed XAIC and FAIC as versions of AIC which correct this bias. We have experimentally shown that these methods give better predictive performance than other methods in many situations.

We see several directions for future work. First, the practical usefulness of our methods needs to be confirmed by further experiments. Other future work includes considering other model selection methods: determining whether they are affected by the same bias that we found for AIC, whether such a bias can be removed (possibly leading to extra-sample and focused versions of those methods), and how these methods perform in simulation experiments and on real data. In particular, BPIC (Ando, 2007) is a promising candidate, as its derivation starts with a Bayesian equivalent of (1). An XBPIC method would also be better able to deal with more complex models than a variant of AIC would have difficulty with, such as hierarchical Bayesian models, greatly increasing its practical applicability.

Acknowledgements

I thank Peter Grünwald and Steven de Rooij for their valuable comments and encouragement.

References

- H. Akaike. Information theory and an extension of the maximum likelihood principle. In B. N. Petrov and F. Csáki, editors, *2nd International Symposium on Information Theory*, pages 267–281, 1973.
- H. Akaike. A Bayesian extension of the minimum AIC procedure of autoregressive model fitting. *Biometrika*, 66(2):237–242, August 1979.
- T. Ando. Bayesian predictive information criterion for the evaluation of hierarchical Bayesian and empirical Bayes models. *Biometrika*, 94(2):443–458, June 2007.
- S. Ben-David, T. Lu, T. Luu, and D. Pál. Impossibility theorems for domain adaptation. In *Artificial Intelligence and Statistics (AISTATS)*, pages 129–136, 2010.
- K. P. Burnham and D. R. Anderson. *Model selection and multimodel inference: A practical information-theoretic approach*. Springer, New York, second edition, 2002.
- G. Claeskens and N. L. Hjort. The focused information criterion. *Journal of the American Statistical Association*, 98:900–916, December 2003. With discussion.
- T. van Erven, P. D. Grünwald, and S. de Rooij. Catching up faster in Bayesian model selection and model averaging. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.
- T. van Erven, P. D. Grünwald, and S. de Rooij. Catching up faster by switching sooner: A predictive approach to adaptive estimation with an application to the AIC-BIC dilemma. *Journal of the Royal Statistical Society, Series B*, 74(3):361–417, 2012. With discussion.
- A. E. Gelfand and S. K. Ghosh. Model choice: A minimum posterior predictive loss approach. *Biometrika*, 85(1):1–11, March 1998.
- G. H. Golub, M. Heath, and G. Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21:215–223, 1979.
- C. M. Hurvich and C-L. Tsai. Regression and time series model selection in small samples. *Biometrika*, 76(2):297–307, June 1989.
- H. Leeb. Evaluation and selection of models for out-of-sample prediction when the sample size is small relative to the complexity of the data-generating process. *Bernoulli*, 14(3):661–690, 2008.
- S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, October 2010.
- G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.
- R. Shibata. Statistical aspects of model selection. In Jan C. Willems, editor, *From data to model*, pages 215–240. Springer-Verlag, 1989.
- D. J. Spiegelhalter, N. G. Best, B. P. Carlin, and A. van der Linde. Bayesian measures of model complexity and fit. *Journal of the Royal Statistical Society, Series B*, 64(4):583–639, 2002. With discussion.
- M. Sugiyama and M. Kawanabe. *Machine learning in non-stationary environments: Introduction to covariate shift adaptation*. MIT Press, Cambridge (MA), 2012.
- M. Sugiyama and H. Ogawa. Subspace information criterion for model selection. *Neural Computation*, 13(8):1863–1889, 2001.
- V. N. Vapnik. *Statistical learning theory*. Wiley, New York, 1998.
- Y. Yang. Can the strenghts of AIC and BIC be shared? *Biometrika*, 92(4):937–950, December 2005.
- Y. Yang. Prediction/estimation with simple linear models: is it really that simple? *Econometric Theory*, 23:1–36, February 2007.

Optimal amortized regret in every interval

Rina Panigrahy
Microsoft Research
Mountain View, CA
rina@microsoft.com

Preyas Popat *
Google Inc.
Mountain View, CA
preyas.p@gmail.com

Abstract

Consider the classical problem of predicting the next bit in a sequence of bits. A standard performance measure is *regret* (loss in payoff) with respect to a set of experts. For example if we measure performance with respect to two constant experts one that always predicts 0's and another that always predicts 1's it is well known that one can get regret $O(\sqrt{T})$ with respect to the best expert by using, say, the weighted majority algorithm [LW89]. But this algorithm does not provide performance guarantee in any interval. There are other algorithms (see [BM07, FSSW97, Vov99]) that ensure regret $O(\sqrt{x \log T})$ in any interval of length x . In this paper we show a randomized algorithm that in an amortized sense gets a regret of $O(\sqrt{x})$ for any interval when the sequence is partitioned into intervals arbitrarily. We empirically estimated the constant in the $O()$ for T upto 2000 and found it to be small – around 2.1. We also experimentally evaluate the efficacy of this algorithm in predicting high frequency stock data.

1 INTRODUCTION

Consider the following classical game of predicting a binary ± 1 sequence. An algorithm A sees a binary sequence $\{b_t\}_{t \geq 1}$, one bit at a time, and attempts to predict the next bit b_t from the past history b_1, \dots, b_{t-1} . The *payoff* A_T of the algorithm in T steps is the number of correct guesses minus the number of the wrong guesses. In other words, let $\tilde{b}_t \in [-1, 1]$ be the prediction for the t^{th} bit based on the previous bits then:

$$A_T := \sum_{1 \leq t \leq T} b_t \tilde{b}_t.$$

The payoff per time step $b_t \tilde{b}_t$ is essentially equivalent to the well known absolute loss function $|b_t - \tilde{b}_t|$ (see for example [CBL06], chapter 8).¹

One can view this game as an idealized “stock prediction” problem as follows. In each unit time, the stock price goes up or down by precisely \$1, and the algorithm bets on this event. If the bet is right, the player wins one dollar, and otherwise loses one dollar. Not surprisingly, in general, it is impossible to guarantee a positive payoff for all possible scenarios (sequences). However, one could hope to give some guarantees on the payoff of the algorithm based on certain properties of the sequence.

For example one can compare the payoff to the better of two choices (experts), which correspond to two constant algorithms: first one, where $\tilde{b}_t = +1$ and the second one where $\tilde{b}_t = -1$ for all t . Note that the best of these experts gets payoff $|\sum_{1 \leq t \leq T} b_t|$, which corresponds to the “optimal in hindsight” expert among the two choices. The *regret* of an algorithm is defined as how much worse the algorithm performs as opposed to the best of the two experts (in hindsight, after seeing the sequence). This has been studied in a number of papers, including

¹since when $|b_t| = 1$, $|b_t - \tilde{b}_t| = |b_t| |b_t - \tilde{b}_t| = |1 - b_t \tilde{b}_t| = 1 - b_t \tilde{b}_t$. Thus the absolute loss function is the negative of our payoff in one step plus a shift of 1. Also b_t values from $\{-1, 1\}$ or $\{0, 1\}$ are equivalent by a simple scaling and shifting transform.

This work was done while this author was at Microsoft Research.

[Cov65, LW89, Cov91, ACBFS02, AB09]. A classical result says that one can obtain a regret of $\Theta(\sqrt{T})$ for a sequence of length T , via, say, the weighted majority algorithm [LW89]. Formally, for a sequence $X = b_1, \dots, b_T$, let $h(X) = \sum_{1 \leq t \leq T} b_t$ denote the “height” of the sequence when plotted cumulatively as a chart. Then we have the following theorem:

Theorem 1.1 [Cov65, CBFH⁺97] *There is an algorithm that achieves payoff $\geq |h(X)| - \alpha\sqrt{T}$. It is also known that the optimal value of $\alpha \rightarrow \sqrt{2/\pi}$ as $T \rightarrow \infty$.*

However, an algorithm that only focuses on the overall regret does not exploit short term trends in the sequence and only relies on a ‘global’ long term bias in the full string. Consider for example a sequence that may not have a high overall bias but has many intervals in which there may be a high level of bias. Our result is that for any partitioning of the sequence into intervals, one can essentially get a regret proportional to \sqrt{x} for each interval of length x in an amortized sense (Theorem 1.3). Although our results are stated for bits they work even when b_t is a real number in $[-1, 1]$. We note that even though similar bounds have been obtained before ([BM07, FSSW97, Vov99] and, more recently, [HS09, KP11]), the penalty on an interval of length x is $O(\sqrt{x \log T})$ in these previous results. Note that in adversarial settings one is interested in a prediction algorithm that can get a positive payoff even if the sequence departs slightly from random; or we may ask what is the smallest amount non-randomness that can be “noticed” by the prediction algorithm. So while our result may seem like just shaving a $\log T$ factor, the reason \sqrt{x} is much better than $\sqrt{x \log T}$ is that in certain adversarial settings (like financial markets), the uptrend or downtrend in total per interval may not be too far from that of a random sequence. Note that a random ± 1 sequence of length x has a height of magnitude $\Theta(\sqrt{x})$ in expectation. So we are saying that even if the height of a sequence of length x is some constant multiple of \sqrt{x} , we get a positive payoff.

The bit prediction problem we consider is closely related to the two experts problem (or multi-armed bandits problem with full information). In each round each expert has a payoff in the range $[0, 1]$ that is unknown to the algorithm. For two experts, let b_{1t}, b_{2t} denote the payoffs of the two experts at time t . The algorithm pulls each arm (expert) with probability $\hat{b}_{1t}, \hat{b}_{2t} \in [0, 1]$ respectively where $\hat{b}_{1t} + \hat{b}_{2t} = 1$. The (expected) payoff of the algorithm in this setting is $A'_T := \sum_{t=1}^T \hat{b}_{1t} b_{1t} + \hat{b}_{2t} b_{2t}$.

We will be concerned with the following payoff function in this paper:

Definition 1.2 (Interval payoff function: P_α)

Let X_1, \dots, X_k denote a partition of the sequence X into a disjoint union of k intervals, that is, X is the concatenation of these k subsequences. We will use $h(X_i)$ to denote the

sum of the bits in the interval X_i and $|X_i|$ to denote the length of X_i .

The interval payoff function, $P_\alpha(X)$ is defined as the maximum value of the expression

$$\sum_{i=1}^k \left(|h(X_i)| - \alpha \sqrt{|X_i|} \right)$$

over all $1 \leq k \leq |X|$ and all partitions X_1, \dots, X_k of X .

We say that a payoff function $f : \{-1, 1\}^T \rightarrow \mathbb{R}$ is feasible if there is a bit prediction algorithm which on sequence X achieves payoff at least $f(X)$.

Theorem 1.3 (Main Theorem) *There is an absolute constant $0 < \alpha < 10$ independent of T such that the interval payoff function P_α is feasible.*

For the two experts problem our result translates to the following guarantee:

$$A'_T \geq \sum_{i=1}^k \left(\max_{j \in \{1, 2\}} \left(\sum_{t \in X_i} b_{jt} \right) - \frac{\alpha}{2} \sqrt{|X_i|} \right).$$

Here $\sum_{t \in X_i} b_{jt}$ is the payoff of the j^{th} expert in the interval X_i .

This result can be viewed as incurring a penalty of $\alpha\sqrt{|X_i|}$ for each interval X_i . We theoretically show that the optimal value of α is at most 10 (Section 2). We empirically estimated the optimal α for T up to 2000 and found it to be small – around 2.1 (Section 4.1).

We stress here that the algorithm doesn’t need to know the partition or the length of the partition in advance. We also note that our guarantee does not hold for each interval individually but when we look at the net payoff in an amortized sense, we may account for a regret of at most $\alpha\sqrt{|X|}$ for an interval of length X . In fact, the guarantee is impossible to achieve in a non-amortized sense. We show that if we measure regret based on the performance of an algorithm in a given interval then one will have to trade-off regrets at different time scales. The following observation is proven in the full version [PP13].

Observation 1.4 *There is no prediction algorithm that can guarantee a regret of $O(\sqrt{|Y|})$ on all intervals Y for all input sequences.*

Regarding the computation of P_α , we show:

Theorem 1.5 *The value of $P_\alpha(S)$ for a particular sequence S of length T can be computed using dynamic programming in time $O(T^3)$.*

For a given T , let $\alpha_0(T)$ denote the minimum α such that P_α is feasible for all sequences of length T . It is possible to

determine α_0 using the following well known observation by Cover.

Observation 1.6 (Cover [Cov65]) A payoff function $f : \{-1, 1\}^T \rightarrow \mathbb{R}$ is feasible if and only if $E_S[f(S)] \leq 0$ where S is a uniformly random sequence in $\{-1, 1\}^T$.

This is achieved by a prediction algorithm that predicts $\tilde{b}_t = \frac{E_U[f(s.1.U)] - E_U[f(s.(-1).U)]}{2}$ where s is the sequence of bits seen so far, U is a suffix sequence chosen uniformly at random and $s.b.U$ denotes the concatenated sequence starting with s followed by bit b followed by the sequence U . Note that $\tilde{b}_t \in [-1, 1]$ as long as for all s , $|E_U[f(s.1.U)] - E_U[f(s.(-1).U)]| \leq 2$

Algorithm and Running time: Theorem 1.5 and Observation 1.6 suggest a simple algorithm for achieving payoff function P_α . Take the sequence s seen so far, append a $+1$ and then a random sequence to make it into a complete sequence of length T . Compute $P_\alpha(S)$ for the resulting sequence S . Do this again replacing the $+1$ by a -1 . Predict \tilde{b}_t to be the half of the difference in the two cases.

We note that a deterministic algorithm achieving the guarantee of Theorem 1.3 may take exponential time since it would need to find $P_\alpha(S)$ for every random completion of the bits seen so far. Alternatively, there is a simple randomized algorithm which achieves the same payoff in expectation by taking a different random completion for every prefix. A naive implementation of this randomized algorithm will take T^3 time for each bit being predicted. We show a simple variant that reduces this to $O(\log T)$ time with pre-computation.

Theorem 1.7 There is a randomized algorithm that achieves the payoff guarantee P_α of Theorem 1.3 in expectation and spends $O(T^2)$ time per step. There is also a randomized algorithm that achieves payoff $P_{\alpha'}$ with $\alpha' = c\alpha$ and spends only $O(\log T)$ time per step. Here $c := \frac{\sqrt{2}}{\sqrt{2}-1}$.

These algorithm use pre-computed information that takes time $O(T^2)$ and $O(T \log T)$ time to compute for the first and the second algorithm respectively.

Generalization to real numbers: In the full version [PP13], we show that a variant of the guarantee holds in a semi-adversarial model where a string of real numbers may be chosen instead of bits. The model combines worst case and average case settings where the signs of the real numbers may be chosen adversarially (that is, in the worst case) but the magnitudes of the real numbers come from a pre-specified distribution independently and randomly.

Experimental results: We implement our algorithm, the weighted majority algorithm, an algorithm based on Autoregressive Integrated Moving Average (ARIMA) and an algorithm of [KP11], and compare their performance when predicting financial time series data. Specifically, we con-

sider the high frequency price data of 5 stocks, and we apply these algorithms to predict the per minute price changes in an online fashion taking the values in each day as a separate sequence. That is we predict the next minute returns of mid-prices for each stock based on its previous 1 minute returns in the day. We perform this experiment over 189 trading days for each stock and find that on an average our algorithm performs better than other prediction algorithms based on regret minimization but is outperformed by the ARIMA algorithm. On the other hand, as we discussed above, our algorithm has certain provable guarantees for every sequence which the ARIMA algorithm lacks. The experimental setup and results are described in more detail in Section 4.

1.1 Related work

There is large body on work on regret style analysis for prediction. Numerous works including [Cov65, CBFH⁺97] have examined the optimal amount of regret achievable with respect to two or more experts. A good reference for the results in this area is [CBL06]. It is well known that in the case of static experts, the optimal regret achievable is exactly equal to the Rademacher complexity of the predictions of the experts (chapter 8 in [CBL06]). Recent works such as [ALW06, AWY08, MS08] have extended this analysis to other settings. Measures other than the standard regret measure have been studied in [RST10]. The question of what can be achieved if one would like to have a significantly better guarantee with respect to a fixed expert or a distribution of experts was asked before in [EDKMW08, KP11]. Tradeoffs between regret and minimum payoff were also examined in [Vov98], where the author studied the set of values of a, b for which an algorithm can have payoff $aOPT + b \log N$, where OPT is the payoff of the best arm and a, b are constants.

Regret minimization algorithms with performance guarantees within each interval have been studied in [BM07, FSSW97, Vov99] and more recently in [HS09, KP11]. As we mentioned, some of these algorithms achieve a regret of $O(\sqrt{x \log T})$ for every interval of size x in a sequence of length T . A related work which also seeks to exploit short term trends in the sequence is [HW98], where the regret bound proportional to \sqrt{Tk} in the best case where k is the number of intervals (see [CBL06], Corollary 5.1). The main difference between the work of [HW98] and our results is that their algorithm requires fixing the number of intervals, k , in advance whereas our algorithm works simultaneously for all k . Also note that their regret guarantee is always higher than the payoff function P_α for a sequence of length T achieving equality only in the special case when all intervals are of equal length T/k .

Numerous papers (for example [Blu97, HSSW98, AHKS06]) have implemented algorithms inspired from

regret style analysis and applied it on financial and other types of data.

1.2 Overview of the proof

In this section we give a high level idea of our proof, the formal proof appears in Section 2.

To prove the main theorem we want to compute the minimum α such that $E_S[P_\alpha(S)] \leq 0$ (See Observation 1.6). We first introduce a variant of the payoff function $P_\alpha(S)$ as follows. Instead of computing the maximum value of $\sum_i |h(X_i)| - \alpha\sqrt{|X_i|}$ over all possible partitions, we will only allow partitions where the intervals are of the form $(2^i j, 2^i(j+1)]$; that is, intervals that are obtained by dividing the string into segments of length that are some power of 2. We will refer to such intervals as ‘aligned’ intervals (Definition 2.3). Further we will only look at T values that is some power of 2. Note that any interval can be broken into at most $\log T$ aligned intervals. Let $P_\alpha^A(S)$ denote the maximum value of $\sum_i |h(X_i)| - \alpha\sqrt{|X_i|}$ with partitions into aligned intervals. We first show that

Lemma 1.8 *If $E[P_\alpha^A(S)] \leq 0$ then $E[P_{c\alpha}(S)] \leq 0$ where $c := \frac{\sqrt{2}}{\sqrt{2}-1}$.*

Next we show

Theorem 1.9 *There is an absolute constant $\alpha \leq 2.8$ such that $E[P_\alpha^A(S)] \leq 0$.*

We prove Theorem 1.9 recursively for T that are increasing powers of 2. We inductively show that the distribution of $P_\alpha^A(S)$ is stochastically upper bounded by a shifted exponential distribution (Definition 2.4) with certain parameters (Equation 2.1), where S is a uniformly random sequence of length T . Since we are dealing with splits into aligned intervals, we can assume that either the best split for S is the whole interval, or the mid-point of S is one of the splitting points. For the first case, we may upper bound the payoff function using Hoeffding’s bound (Theorem 2.2), while for the second case we may inductively assume that the distribution of payoffs for the subsequences is stochastically bounded by a shifted exponential distribution. We then separately bound each of these distributions by the shifted exponential distribution.

2 FEASIBILITY OF PAYOFF FUNCTION P_α

2.1 Preliminaries

Definition 2.1 (Binomial distribution B_n) *Let $x_1, x_2, \dots, x_n \in \{-1, 1\}$ be uniformly and independently distributed. Then the sum*

$$Y := \sum_{i=1}^n x_i$$

is said to be binomially distributed. We denote the distribution as B_n .

Theorem 2.2 (Hoeffding’s bound) [Hoe63]

$$\Pr[|B_n| \geq y \cdot \sqrt{n}] \leq 2 \cdot \exp\left(-\frac{y^2}{2}\right)$$

Definition 2.3 (Aligned interval)

We assume here that T is a power of 2. An aligned interval is one which is obtained by breaking $[1, T]$ into 2^i equal parts for $i \in [0, \log T]$ and picking one of the parts. So for instance the first part is always $[1, T/2^i]$; each aligned interval can be written as $[jT/2^i + 1, (j+1)T/2^i]$ for non negative integers i and j .

We denote the interval payoff function corresponding to Definition 1.2 which allows only aligned splits as P_α^A .

Definition 2.4 (Shifted Exponential distribution) *The probability density function $f_{\mu, \sigma, n}$ of shifted exponential distribution with mean $\sigma\sqrt{n}$ and shift $\mu\sqrt{n}$ is defined as follows:*

$$\begin{aligned} f_{\mu, \sigma, n}(y) &:= \frac{1}{\sigma\sqrt{n}} \exp\left(-\frac{y - \mu\sqrt{n}}{\sigma\sqrt{n}}\right) & \forall y \geq \mu\sqrt{n} \\ f_{\mu, \sigma, n}(y) &:= 0 & \forall y \leq \mu\sqrt{n} \end{aligned}$$

We denote a random variable distributed according to $f_{\mu, \sigma, n}$ as $F_{\mu, \sigma, n}$. That is, $\Pr[F_{\mu, \sigma, n} \geq y] = \int_y^\infty f_n(s) ds = \exp\left(-\frac{y - \mu\sqrt{n}}{\sigma\sqrt{n}}\right)$ when $y \geq \mu\sqrt{n}$ and 1 otherwise.

2.2 Proof of feasibility

The following lemma is a restatement of lemma 1.8 and is proven using a standard doubling trick

Lemma 2.5 *If P_α^A is feasible then $P_{c\alpha}$ is also feasible, where $c := \frac{\sqrt{2}}{\sqrt{2}-1}$.*

Proof:

Let X_1, X_2, \dots, X_k denote a partition of a given sequence S . We split each interval X_i into a disjoint union of aligned intervals Y_{i1}, \dots, Y_{il} . We will then show that the identity

$$\sum_{j=1}^l \sqrt{|Y_{ij}|} \leq c \cdot \sqrt{|X_i|}$$

always holds where $|I|$ denotes the length of the interval I . This suffices to prove the theorem since $h(X_i) \leq \sum_{j=1}^l h(Y_{ij})$.

For notational simplicity, let $I = X_i$ and $x = |I|$. If I is an aligned interval we are done, otherwise we write it as the minimal union of aligned intervals (take out the largest aligned interval in I and repeat). There are three possibilities:-

1. $I = I_1 \cup I_2$ is a union of two intervals of size $x/2$ each (eg. the interval $[T/4 + 1, 3T/4]$)
2. $I = I_1 \cup I_2 \cup \dots \cup I_l$, where each I_j is of a different size. Note that all interval sizes on the right are powers of 2 and strictly less than x
3. $I = J \cup J'$ where each J can be written as a union of intervals as in 1 or 2 above

In the first case,

$$\sqrt{|I_1|} + \sqrt{|I_2|} \leq 2 \cdot \sqrt{x/2} = \sqrt{2} \cdot \sqrt{x}$$

In the second case,

$$\sum_{j=1}^l \sqrt{|I_j|} \leq \sqrt{x} \cdot \sum_{j=1}^{\infty} \sqrt{1/2^j} = \frac{1}{\sqrt{2}-1} \cdot \sqrt{x}$$

In the third case,

$$\begin{aligned} \sqrt{|J|} + \sqrt{|J'|} &\leq \frac{1}{\sqrt{2}-1} \cdot \sqrt{|J|} + \\ &\frac{1}{\sqrt{2}-1} \cdot \sqrt{|J'|} \leq \frac{\sqrt{2}}{\sqrt{2}-1} \cdot \sqrt{x} \end{aligned}$$

■

We are now ready to prove Theorem 1.9.

Proof: [Proof of Theorem 1.9] We need to show that for all $T \geq 1$, $\mathbb{E}_{x \in \{-1,1\}^T} [P_\alpha^A(x)] \leq 0$. After that, the theorem follows from Observation 1.6 (it is easy to check that the condition required for required for $\tilde{b}_t \in [-1,1]$ given in Observation is satisfied by P_α^A).

We will prove the theorem by induction. We will show that when n is a power of 2,

$$\forall y \in \mathbb{R} \quad \Pr_{x \in \{-1,1\}^n} [P_\alpha^A(x) \geq y] \leq \Pr[F_{\mu,\sigma,n} \geq y] \quad (2.1)$$

for some $\mu := \mu(\alpha)$ and $\sigma := \sigma(\alpha)$. Here $F_{\mu,\sigma,n}$ is as in Definition 2.4.

Note that this would imply $\mathbb{E}_{x \in \{-1,1\}^n} [P_\alpha^A(x)] \leq \mathbb{E}[F_{\mu,\sigma,n}] = (\mu + \sigma)\sqrt{n}$. We will show that for a suitable choice of α , the term $\mu + \sigma \leq 0$, and this suffices to prove the theorem.

It remains to prove Equation 2.1. For the base case, $n = 1$, we see that the equation is satisfied for $\mu \geq 1 - \alpha$, $\sigma > 0$. We will now show that it is satisfied for $2n$ whenever it is satisfied for n (for appropriate μ and σ).

Now, for a sequence $x := (x_1, x_2) \in \{-1,1\}^n \times \{-1,1\}^n$, $P_\alpha^A(x) = \max(P_\alpha^A(x_1) + P_\alpha^A(x_2), |h(x)| - \alpha \cdot \sqrt{2n})$. So for every x such that $P_\alpha^A(x) \geq y$ we must have either $P_\alpha^A(x_1) + P_\alpha^A(x_2) \geq y$ or that $h(x) - \alpha \cdot \sqrt{2n} \geq y$. Thus,

$$\Pr_{x \in \{-1,1\}^{2n}} [P_\alpha^A(x) \geq y] \quad (2.2)$$

$$\leq \Pr_{x_1, x_2 \in \{-1,1\}^n} [P_\alpha^A(x_1) + P_\alpha^A(x_2) \geq y] \quad (2.3)$$

$$+ \Pr_{x \in \{-1,1\}^{2n}} [h(x) - \alpha \cdot \sqrt{2n} \geq y] \quad (2.4)$$

$$\leq \Pr[F_{\mu,\sigma,n} + F'_{\mu,\sigma,n} \geq y] \quad (2.5)$$

$$+ \Pr_{x \in \{-1,1\}^{2n}} [h(x) - \alpha \cdot \sqrt{2n} \geq y] \quad (2.6)$$

Here F and F' are independent random variables distributed as in Definition 2.4. We will show that the first and second term are each bounded by $\frac{1}{2} \Pr[F_{2n} \geq y]$ which is sufficient to prove Equation 2.1. Note that we only need to consider $y \geq \mu\sqrt{2n}$ since for smaller values of y we have

$$\Pr_{x \in \{-1,1\}^{2n}} [P_\alpha^A(x) \geq y] \leq \Pr[F_{2n} \geq y] = 1$$

Henceforth, we will use shorthands $f_n := f_{\mu,\sigma,n}$ and $F_n := F_{\mu,\sigma,n}$.

The first term can be written as:-

$$\begin{aligned} \Pr[F_n + F'_n \geq y] &= \int_y^\infty \int_{-\infty}^\infty f_n(s) \cdot f_n(w-s) \, ds \, dw \\ &= \int_y^\infty \int_{\mu\sqrt{n}}^{w-\mu\sqrt{n}} f_n(s) \cdot f_n(w-s) \, ds \, dw \end{aligned}$$

where the second equation follows from the fact that $f_n(s) = 0$ for $s < \mu\sqrt{n}$ and $f_n(w-s) = 0$ for

$s > w - \mu\sqrt{n}$. Thus, we need to show for all $y \geq \mu\sqrt{2n}$:

$$\begin{aligned}
& \int_y^\infty \int_{\mu\sqrt{n}}^{w-\mu\sqrt{n}} f_n(s) \cdot f_n(w-s) \, ds \, dw \leq \frac{1}{2} \Pr[F_{2n} \geq y] \\
& \Leftarrow \frac{1}{\sigma^2 n} \int_y^\infty \int_{\mu\sqrt{n}}^{w-\mu\sqrt{n}} \exp\left(-\frac{w+2\mu\sqrt{n}}{\sigma\sqrt{n}}\right) \, ds \, dw \\
& \leq \frac{1}{2} \exp\left(-\frac{y-\mu\sqrt{2n}}{\sigma\sqrt{2n}}\right) \\
& \Leftarrow \frac{1}{\sigma^2 n} \int_y^\infty \int_{\mu\sqrt{n}}^{w-\mu\sqrt{n}} \exp\left(-\frac{w-2\mu\sqrt{n}}{\sigma\sqrt{n}}\right) \, ds \, dw \\
& \leq \frac{1}{2} \exp\left(-\frac{y-\mu\sqrt{2n}}{\sigma\sqrt{2n}}\right) \\
& \Leftarrow \frac{1}{\sigma^2 n} \int_y^\infty (w-2\mu\sqrt{n}) \exp\left(-\frac{w-2\mu\sqrt{n}}{\sigma\sqrt{n}}\right) \, dw \\
& \leq \frac{1}{2} \exp\left(-\frac{y-\mu\sqrt{2n}}{\sigma\sqrt{2n}}\right)
\end{aligned}$$

In the third line we implicitly assume that $y \geq 2\mu\sqrt{n}$, since otherwise the left hand side is less than 0 and the equation is satisfied.

Note that the integral is of the form $\int u \cdot e^{-cu}$ which integrates to $-\left(\frac{u+1/c}{c}\right) \cdot e^{-cu}$. Thus, integrating and substituting $z := y - 2\mu\sqrt{n}$ we need to show for all $z \geq 0$,

$$\begin{aligned}
& \frac{1}{\sigma\sqrt{n}} \cdot (z + \sigma\sqrt{n}) \cdot \exp\left(-\frac{z}{\sigma\sqrt{n}}\right) \\
& \leq \frac{1}{2} \exp\left(-\frac{z + (\sqrt{2}-1)\mu\sqrt{2n}}{\sigma\sqrt{2n}}\right) \\
& \Leftarrow \frac{2z}{\sigma\sqrt{n}} + 2 \\
& \leq \exp\left(\frac{z}{\sigma\sqrt{n}} - \frac{z + (\sqrt{2}-1)\mu\sqrt{2n}}{\sigma\sqrt{2n}}\right) \\
& \Leftarrow \frac{2z}{\sigma\sqrt{n}} + 2 \\
& \leq \exp\left(\frac{(\sqrt{2}-1)z}{\sigma\sqrt{2n}}\right) \cdot \exp\left((\sqrt{2}-1)\frac{-\mu}{\sigma}\right)
\end{aligned}$$

Substituting $w := \frac{z}{\sigma\sqrt{n}}$, we need for all $w \geq 0$,

$$\begin{aligned}
& 2w + 2 \\
& \leq \exp\left(\frac{(\sqrt{2}-1)w}{\sqrt{2}}\right) \cdot \exp\left((\sqrt{2}-1)\frac{-\mu}{\sigma}\right) \\
& \Leftarrow \frac{2w+2}{\exp\left(\frac{(\sqrt{2}-1)w}{\sqrt{2}}\right)} \leq \exp\left((\sqrt{2}-1)\frac{-\mu}{\sigma}\right)
\end{aligned}$$

The left hand side is maximized at $w = 1/\sqrt{2}$ and the value of left hand side at that point is around 2.78. Thus, if $(-\mu/\sigma) \geq 2.47$ then the equation is always satisfied.

We now turn to bounding the second term 2.6. We need to show for all $y \geq \mu\sqrt{2n}$,

$$\begin{aligned}
& \Pr_{x \in \{-1,1\}^{2n}}[|x| - \alpha \cdot \sqrt{2n} \geq y] \leq \frac{1}{2} \Pr[F_{2n} \geq y] \\
& \Leftarrow \Pr[|B_{2n}| \geq y + \alpha \cdot \sqrt{2n}] \leq \frac{1}{2} \Pr[F_{2n} \geq y] \\
& \Leftarrow \Pr[|B_{2n}| \geq (z + \alpha) \cdot \sqrt{2n}] \leq \frac{1}{2} \Pr[F_{2n} \geq z \cdot \sqrt{2n}] \\
& \Leftarrow 2 \cdot \exp\left(-\frac{(z + \alpha)^2}{2}\right) \leq \frac{1}{2} \Pr[F_{2n} \geq z \cdot \sqrt{2n}]
\end{aligned}$$

where the last line follows from Theorem 2.2, and in the second last line we substitute $z := y/\sqrt{2n}$.

Thus, we need to show for all $z \geq \mu$,

$$4 \cdot \exp\left(-\frac{(z + \alpha)^2}{2}\right) \leq \exp\left(-\frac{z\sqrt{2n} - \mu\sqrt{2n}}{\sigma\sqrt{2n}}\right)$$

Substituting $w := z - \mu$, we need to show for all $w \geq 0$,

$$\begin{aligned}
& \exp\left(-\frac{(w + \mu + \alpha)^2}{2} + \frac{w}{\sigma}\right) \leq 0.25 \\
& \Leftarrow -\frac{(w + \mu + \alpha)^2}{2} + \frac{w}{\sigma} \leq -1.4
\end{aligned}$$

The left hand side is maximized at $w + \mu + \alpha = 1/\sigma$ and for that value of w the inequality is given by

$$-\frac{1}{2\sigma^2} + \frac{1/\sigma - \mu - \alpha}{\sigma} \leq -1.4 \Leftarrow \mu + \alpha \geq 1.4\sigma + \frac{0.5}{\sigma}$$

Also, recall that to bound the first term we needed $-\frac{\mu}{\alpha} \geq 2.47$. Let's set $\mu := -2.47\sigma$. Then we need

$$\alpha \geq (1.4 + 2.47)\sigma + \frac{0.5}{\sigma} = 3.87\sigma + \frac{0.5}{\sigma}$$

The right hand side is minimized at $\sigma = \frac{1}{\sqrt{2 \cdot 3.87}} \approx 0.36$, and substituting we get that $\alpha = 2.8$ is feasible. Recall that we also needed $\mu + \alpha \geq 1$ from the base case which is already satisfied for this choice of parameters.

3 ALGORITHM AND RUNNING TIME

We will now prove theorem 1.5.

Proof: [Proof of Theorem 1.5] We give a simple $O(T^2)$ space and $O(T^3)$ time algorithm.

For every subinterval (i, j) of the sequence, $i, j \in [T]$ the DP table stores $P_\alpha(S_{ij})$ where S_{ij} is the subsequence of S containing bits from position i to position j , inclusive. For $i = j$, this value is always $1 - \alpha$. For $j > i$, to compute the value of $P_\alpha(S_{ij})$, we need to take the maximum over two quantities. The first quantity is $|h(S_{ij})| - \alpha \cdot \sqrt{j - i + 1}$ which corresponds to splitting the subsequence into a single interval. This can be readily computed in constant time if we pre-compute the height of every subsequence, which can be done in $O(T^2)$ space and time. The second quantity is the maximum over all $k \in \{i, i + 1, \dots, j\}$ of $P_\alpha(S_{ik}) + P_\alpha(S_{kj})$. This corresponds to splitting the subsequence at k and then recursively computing the best payoff in each of the two intervals created. This quantity can be computed in time $j - i + 1$ since for each k we just need to read off the appropriate values ($P_\alpha(S_{ik})$ and $P_\alpha(S_{kj})$) from the DP table. ■

Next we prove Theorem 1.7

Proof: [Proof of Theorem 1.7]

Let $X \in \{-1, 1\}^T$ be the input sequence we are required to predict. Using Observation 1.6, it is easy to see that the following algorithm achieves payoff $P_\alpha(X)$ in expectation. For every $t \in \{0, 1, \dots, T - 1\}$:

1. Let $s \in \{-1, 1\}^t$ be the sequence of bits seen so far.
2. Let U_t be a sequence drawn uniformly at random from $\{-1, 1\}^{T-t-1}$ (independently for each t). Let $s_1 := s \cdot 1 \cdot U$ and $s_{-1} := s \cdot (-1) \cdot U$.
3. Make the prediction $\tilde{b} := (P_\alpha(s_1) - P_\alpha(s_{-1}))/2$ for the next bit.

The key idea is that we will draw a random sequence of length T and use its suffix of length $\{-1, 1\}^{T-t-1}$ as U_t . In advance we pre-compute enough information to make the

prediction as fast as possible. For each $t \in \{0, 1, \dots, T - 1\}$ we pre-compute the following information for each U_t :

1. $h(U_t^1)$ for every prefix U_t^1 of U_t
2. $P_\alpha(U_t^2)$ for every suffix U_t^2 of U_t

The pre-computation takes $O(T^2)$ time as P_α is computed only for each suffix.

Let's describe how to use this pre-computed information to compute $P_\alpha(s_1)$ at time t (the computation of $P_\alpha(s_{-1})$ is similar). Let $1 \leq i \leq t$ and $t + 2 \leq j \leq T$. Then it is easy to check that

$$P_\alpha = \max_{i,j} (P_\alpha(s_{1i}) + P_\alpha(U_{jt})) + |h(s_{(i+1)t})| + |h(U_{(t+1)(j-1)})| - \alpha \cdot \sqrt{j - i - 1}$$

Here for a sequence S , S_{ij} is the subsequence of S containing bits from position i to position j , inclusive. Note that we think of U_t as being indexed from $t + 1$ to T where the $(t + 1)^{th}$ bit is 1 (since we are dealing with s_1). The second and fourth term are part of our pre-computation. The first and third terms can be computed on the fly and stored in the table as we increase t from 1 to T . Thus, for each i and j we can compute this expression in constant time and hence we can produce a prediction in $O(T^2)$ time per step.

The second part of the theorem is proved in a similar manner by using only aligned intervals for splitting the sequence (Definition 2.3) and observing that the number of aligned intervals spanning a given position is at most $O(\log T)$. The algorithm achieves payoff at least $P_{\alpha'}$ because of Lemma 1.8. ■

4 EXPERIMENTAL RESULTS

In this section we describe our experimental setup and findings.

The first part of the experiment is to experimentally estimate the value of α_0 . In general we may think of α_0 as a function of T . In Section 2 we saw that $\alpha_0(T)$ is bounded from above by an absolute constant for all T . In Section 4.1 below we estimate the values of α_0 for a range of T .

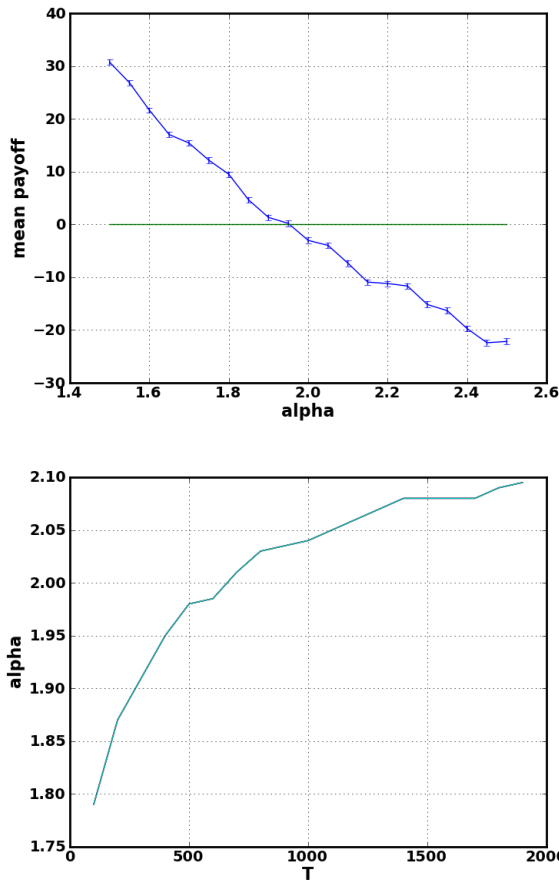
The second part of the experiment is to implement our algorithm and compare its performance against 3 other prediction algorithms. This is described in Section 4.2 below.

4.1 Computation of α_0

We denote by $\alpha_0(T)$ the minimum value of α such that the payoff function P_α is feasible for sequences of length T . For a particular T , this value can be computed using Theorem 1.5. While Theorem 1.5 requires us to compute the

payoff function over all sequences of length T (to compute the expectation), we can experimentally approximate this by taking sufficiently many random sequences of length T and looking at the expectation of the sample. We are interested in $T = 389$ which is the number of minutes in a trading day for which we have returns data (there are 390 minutes in a typical trading day and the returns for the first minute is undefined).

Note that the standard error of the sample mean is obtained as the sample standard deviation divided by \sqrt{n} where $n = 400$ is the number of trials. The chart on the left shows the mean payoff and standard error for various values of α for $T = 389$.



From the figure we see that $\alpha = 1.96$ is a good estimate for $\alpha_0(T)$ for $T = 389$. The figure on the right shows estimated values of α_0 for various T .

4.2 Comparison of predictive performance

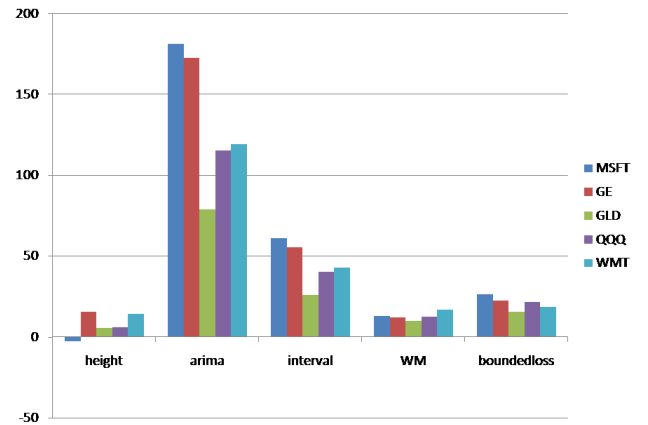
The algorithms we consider are:-

1. The baseline buy and hold strategy that achieves payoff equal to the height (height)
2. The algorithm described in this paper (interval)

3. Weighted Majority algorithm (WM)
4. The algorithm of [KP11] (Algorithm 4, section 5) (boundedloss)
5. An algorithm based on Auto Regressive Integrated Moving Average (arima)

Note that algorithms 2-4 are based on ideas from regret minimization with provable guarantees while the fifth is a commonly used model for predicting time series data. To implement the fourth algorithm we use the function `AUTO.ARIMA()` in R which is part of the library `FORECAST`.

The prediction task we consider is to predict the next minute returns for a stock over a single trading day using only the previous 1 minute returns of the given stock for the given day. More precisely, we define the price of a stock at a given time taking the average of the best bid price and best ask price at that time as reported by the New York Stock Exchange (NYSE). We perform this prediction experiment over 189 days for the following 5 US stocks/ETFs from various sectors: MSFT, GE, GLD, QQQ and WMT. This gives us performance data for each algorithm for a total of $389 \times 189 \times 5 = 367,605$ data points. The results obtained are shown in the figure below.



We note that while our algorithm performs better in practice than other regret minimization based prediction algorithms with provable guarantees, it is outperformed by the ARIMA model.

References

- [AB09] J.-Y. Audibert and S. Bubeck. Minimax policies for adversarial and stochastic bandits. *COLT*, 2009. 1
- [ACBFS02] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. Schapire. The nonstochastic multi-armed

- bandit problem. *SIAM J. Comput.*, 32:48–77, 2002. 1
- [AHKS06] A. Agarwal, E. Hazan, S. Kale, and R.E. Schapire. Algorithms for portfolio management based on the newton method. In *Proceedings of the 23rd international conference on Machine learning*, pages 9–16. ACM, 2006. 1.1
- [ALW06] J. Abernethy, J. Langford, and M. Warmuth. Continuous experts and the binning algorithm. *Learning Theory*, pages 544–558, 2006. 1.1
- [AWY08] J. Abernethy, M.K. Warmuth, and J. Yellin. Optimal strategies from random walks. In *Proceedings of The 21st Annual Conference on Learning Theory*, pages 437–446. Cite-seer, 2008. 1.1
- [Blu97] A. Blum. Empirical support for winnow and weighted-majority algorithms: Results on a calendar scheduling domain. *Machine Learning*, 26(1):5–23, 1997. 1.1
- [BM07] A. Blum and Y. Mansour. From external to internal regret. *Journal of Machine Learning Research*, pages 1307–1324, 2007. (document), 1, 1.1
- [CBFH⁺97] N. Cesa-Bianchi, Y. Freund, D. Haussler, D.P. Helmbold, R.E. Schapire, and M.K. Warmuth. How to use expert advice. *Journal of the ACM (JACM)*, 44(3):427–485, 1997. 1.1, 1.1
- [CBL06] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning and Games*. Cambridge University Press, 2006. 1, 1.1
- [Cov65] T. Cover. Behaviour of sequential predictors of binary sequences. *Transactions of the Fourth Prague Conference on Information Theory, Statistical Decision Functions, Random Processes*, 1965. 1, 1.1, 1.6, 1.1
- [Cov91] T. Cover. Universal portfolios. *Mathematical Finance*, 1991. 1
- [EDKMW08] E. Even-Dar, M. Kearns, Y. Mansour, and J. Wortman. Regret to the best vs. regret to the average. *Machine Learning*, 72:21–37, 2008. 1.1
- [FSSW97] Y. Freund, R. E. Schapire, Y. Singer, and M. K. Warmuth. Using and combining predictors that specialize. *STOC*, pages 334–343, 1997. (document), 1, 1.1
- [Hoe63] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963. 2.2
- [HS09] E. Hazan and C. Seshadhri. Efficient learning algorithms for changing environments. *ICML*, pages 393–400, 2009. 1, 1.1
- [HSSW98] D.P. Helmbold, R.E. Schapire, Y. Singer, and M.K. Warmuth. On-line portfolio selection using multiplicative updates. *Mathematical Finance*, 8(4):325–347, 1998. 1.1
- [HW98] Mark Herbster and Manfred K Warmuth. Tracking the best expert. *Machine Learning*, 32(2):151–178, 1998. 1.1
- [KP11] Michael Kapralov and Rina Panigrahy. Prediction strategies without loss. In *NIPS*, pages 828–836 (full version is available at <http://arxiv.org/abs/1008.3672>), 2011. 1, 1.1, 4
- [LW89] N. Littlestone and M.K. Warmuth. The weighted majority algorithm. *FOCS*, 1989. (document), 1
- [MS08] I. Mukherjee and R. Schapire. Learning with continuous experts using drifting games. In *Algorithmic Learning Theory*, pages 240–255. Springer, 2008. 1.1
- [PP13] Rina Panigrahy and Preyas Papat. Optimal amortized regret in every interval. *arXiv preprint arXiv:1304.7577*, 2013. 1, 1
- [RST10] A. Rakhlin, K. Sridharan, and A. Tewari. Online learning: Beyond regret. *arXiv preprint arXiv:1011.3168*, 2010. 1.1
- [Vov98] V. Vovk. A game of prediction with expert advice. *Journal of Computer and System Sciences*, 1998. 1.1
- [Vov99] V. Vovk. Derandomizing stochastic prediction strategies. *Machine Learning*, pages 247–282, 1999. (document), 1, 1.1

Learning Partial Policies to Speedup MDP Tree Search

Jervis Pinto

School of EECS
Oregon State University
Corvallis OR 97331 USA
pinto@eecs.oregonstate.edu

Alan Fern

School of EECS
Oregon State University
Corvallis OR 97331 USA
afern@eecs.oregonstate.edu

Abstract

A popular approach for online decision making in large MDPs is time-bounded tree search. The effectiveness of tree search, however, is largely influenced by the action branching factor, which limits the search depth given a time bound. An obvious way to reduce action branching is to consider only a subset of potentially good actions at each state as specified by a provided partial policy. In this work, we consider offline learning of such partial policies with the goal of speeding up search without significantly reducing decision-making quality. Our first contribution is to study learning algorithms based on reducing our learning problem to i.i.d. supervised learning. We give a reduction-style analysis of three such algorithms, each making different assumptions, which relates the supervised learning objectives to the sub-optimality of search using the learned partial policies. Our second contribution is to describe concrete implementations of the algorithms within the popular framework of Monte-Carlo tree search. Finally, the third contribution is to evaluate the learning algorithms in two challenging MDPs with large action branching factors, showing that the learned partial policies can significantly improve the anytime performance of Monte-Carlo tree search.

1 INTRODUCTION

Lookahead tree search is a common approach for time-bounded decision making in large Markov Decision Processes (MDPs). Actions are selected by estimating action values at the current state by building a finite-horizon lookahead tree using an MDP model or simulator. A variety of algorithms are available for building such trees, including instances of Monte-Carlo Tree Search (MCTS) such as UCT (Kocsis and Szepesvári, 2006), Sparse Sam-

pling (Kearns et al., 2002), and FSSS (Walsh et al., 2010), along with model-based search approaches such as RTDP (Barto et al., 1995) and AO* (Bonet and Geffner, 2012). Given time constraints, however, the performance of these approaches depends on the action branching factor, which is often considerable and greatly limits the feasible search depth. An obvious way to address this problem is to provide prior knowledge for explicitly pruning bad actions from consideration. In this paper, we consider offline learning of such prior knowledge in the form of partial policies.

A partial policy is a function that quickly returns an action subset for each state and can be integrated into search by pruning away actions not included in the subsets. Thus, a partial policy can significantly speedup search if it returns small action subsets, provided that the overhead for applying the partial policy is small enough. If those subsets typically include high-quality actions, then we might expect little decrease in decision-making quality. Although learning partial policies to guide tree search is a natural idea, it has received surprisingly little attention, both in theory and practice. In this paper we formalize this learning problem from the perspective of “speedup learning”. We are provided with a distribution over search problems in the form of a root state distribution and a search depth bound. The goal is to learn partial policies that significantly speedup depth D search, while bounding the expected regret of selecting actions using the pruned search versus no pruning.

In order to solve this learning problem, there are at least two key choices that must be made: 1) Selecting a training distribution over states arising in lookahead trees, and 2) Selecting a loss function that the partial policy is trained to minimize with respect to the chosen distribution. The key contribution of our work is to consider a family of reduction-style algorithms that answer these questions in different ways. In particular, we consider three algorithms that reduce partial policy learning to i.i.d. supervised learning problems characterized by choices for (1) and (2). Our main results bound the sub-optimality of tree search using the learned partial policies in terms of the expected loss of the supervised learning problems. Interestingly, these re-

sults for learning partial policies mirror similar reduction-style results for learning (complete) policies via imitation, e.g. (Ross and Bagnell, 2010; Syed and Schapire, 2010; Ross et al., 2011).

We empirically evaluate our algorithms in the context of learning partial policies to speedup MCTS in two challenging domains with large action branching factors: 1) the classic dice game, Yahtzee, and 2) a real-time strategy game, Galcon. The results show that using the learned partial policies to guide MCTS leads to significantly improved anytime performance in both domains. Furthermore, we show that several other existing approaches for injecting knowledge into MCTS are not as effective as using partial policies for action pruning and can often hurt search performance rather than help.

2 PROBLEM SETUP

We consider sequential decision making in the framework of Markov Decision Processes (MDPs) and assume basic familiarity. An MDP is a tuple (S, A, P, R) , where S is a finite set of states, A a finite set of actions, $P(s'|s, a)$ is the transition probability of arriving at state s' after executing action a in state s , and $R(s, a) \in [0, 1]$ is the reward function giving the reward of taking action a in state s . The typical goal of MDP planning and learning is to compute a policy for selecting an action in any state, such that following the policy (approximately) maximizes some measure of long-term expected reward.

In practice, regardless of the long-term reward measure, for large MDPs, the offline computation of high-quality policies over all environment states is impractical. In such cases, a popular action-selection approach is online tree search, where at each encountered environment state, a time-bounded search is carried out in order to estimate action values. Note that this approach requires the availability of either an MDP model or an MDP simulator in order to construct search trees. In this paper, we assume that a model or simulator is available and that online tree search has been chosen as the action selection mechanism. Next, we formally describe the paradigm of online tree search, introduce the notion of partial policies for pruning tree search, and then formulate the problem of offline learning of such partial policies.

Online Tree Search. We will focus on depth-bounded search assuming a search depth bound of D throughout, which bounds the length of future action sequences to be considered. Given a state s , we denote by $T(s)$ the depth D expectimax tree rooted at s . $T(s)$ alternates between layers of state nodes and action nodes, labeled by states and actions respectively. The children of each state node are action nodes for each action in A . The children of an action node a with parent labeled by s are all states s' such that $P(s'|s, a) > 0$. Figure 1(a) shows an example of a

depth two expectimax tree. The depth of a state node is the number of action nodes traversed from the root to reach it, noting that leaves of $T(s)$ will always be action nodes.

The optimal value of a state node s at depth d , denoted by $V_d^*(s)$, is equal to the maximum value of its child action nodes, which we denote by $Q_d^*(s, a)$ for child a . We define $Q_d^*(s, a)$ to be $R(s, a)$ if $d = D - 1$ (i.e. for leaf action nodes) and otherwise $R(s, a) + E[V_{d+1}^*(s')]$, where $s' \sim P(\cdot|s, a)$ ranges over children of a . The optimal action policy for state s at depth d will be denoted by $\pi_d^*(s) = \arg \max_a Q_d^*(s, a)$. Given an environment state s , online search algorithms such as UCT or RTDP attempt to completely or partially search $T(s)$ in order to approximate the root action values $Q_0^*(s, a)$ well enough to identify the optimal action $\pi_0^*(s)$. It is important to note that optimality in our context is with respect to the specified search depth D , which may be significantly smaller than the expected planning horizon in the actual environment. This is a practical necessity that is often referred to as receding-horizon control. Here we simply assume that an appropriate search depth D has been specified and our goal is to speedup planning within that depth.

Search with Partial Policies. One way to speedup depth D search is to prune actions from $T(s)$. In particular, if a fixed fraction σ of actions are removed from each state node, then the size of the tree would decrease by a factor of $(1 - \sigma)^D$, potentially resulting in significant computational savings. For this purpose we will utilize partial policies. A depth D (non-stationary) partial policy ψ is a sequence $(\psi_0, \dots, \psi_{D-1})$ where each ψ_d maps a state to an action subset. Given a partial policy ψ and root state s , we can define a pruned tree $T_\psi(s)$ that is identical to $T(s)$, except that each state s at depth d only has subtrees for actions in $\psi_d(s)$, pruning away subtrees for any child $a \notin \psi_d(s)$. Figure 1(b) shows a pruned tree $T_\psi(s)$, where ψ prunes away one action at each state. It is straightforward to incorporate ψ into a search algorithm by only expanding actions at state nodes that are consistent with ψ .

We define the state and action values relative to $T_\psi(s)$ the same as above and let $V_d^\psi(s)$ and $Q_d^\psi(s, a)$ denote the depth d state and action value functions. We will denote the highest-valued, or greedy, root action of $T_\psi(s)$ by $G_\psi(s) = \arg \max_{a \in \psi_0(s)} Q_0^\psi(s, a)$. This is the root action that a depth D search procedure would attempt to return in the context of ψ . Note that a special case of partial policies is when $|\psi_d(s)| = 1$ for all s and d , which means that ψ defines a traditional (complete) deterministic MDP policy. In this case, V_d^ψ and Q_d^ψ represent the traditional depth d state value and action value functions for policies. We say that a partial policy ψ subsumes a partial policy ψ' if for each s and d , $\psi'_d(s) \subset \psi_d(s)$. In this case, it is straightforward to show that for any s and d , $V_d^{\psi'}(s) \leq V_d^\psi(s)$.

Clearly, a partial policy can reduce the complexity of

search. However, we are also concerned with the quality of decision making using $T_\psi(s)$ versus $T(s)$, which we will quantify in terms of expected regret. The regret of selecting action a at state s relative to $T(s)$ is equal to $V_0^*(s) - Q_0^*(s, a)$, noting that the regret of the optimal action $\pi_0^*(s)$ is zero. We prefer partial policies that result in root decisions with small regret over the root state distribution that we expect to encounter, while also supporting significant pruning. For this purpose, if μ_0 is a distribution over root states, we define the expected regret of ψ with respect to μ_0 as $\text{REG}(\mu_0, \psi) = E[V_0^*(s_0) - Q_0^*(s_0, G_\psi(s_0))]$, where $s_0 \sim \mu_0$.

Learning Problem. We consider an offline learning setting where we are provided with a model or simulator of the MDP in order to train a partial policy that will be used later for online decision making. That is, the learning problem provides us with a distribution μ_0 over root states (or a representative sample from μ_0) and a depth bound D . The intention is for μ_0 to be representative of the states that will be encountered during online use. In this work, we are agnostic about how μ_0 is defined for an application. However, a typical example and one used in our experiments, is for μ_0 to correspond to the distribution of states encountered along trajectories of a receding horizon controller that makes decisions based on unpruned depth D search.

Given μ_0 and D , our “speedup learning” goal is to learn a partial policy ψ with small expected regret $\text{REG}(\mu_0, \psi)$, while providing significant pruning. That is, we want to approximately imitate the decisions of depth D unpruned search via a much less expensive depth D pruned search. In general, there will be a tradeoff between the potential speedup and expected regret. At one extreme, it is always possible to achieve zero expected regret by selecting a partial policy that does no pruning and hence no speedup. At the other extreme, we can remove the need for any search by learning a partial policy that always returns a single action (i.e. a complete policy). However, for many complex MDPs, it can be difficult to learn computationally efficient, or reactive, policies that achieve small regret. Rather, it may be much easier to learn partial policies that prune away many, but not all actions, yet still retain high-quality actions. While such partial policies lead to more search than a reactive policy, the regret can be much less.

In practice, we seek a good tradeoff between the two extremes, which will depend on the application. Instead of specifying a particular trade-off point as our learning objective, we develop learning algorithms in the next section that provide some ability to explore different points. In particular, the algorithms are associated with regret bounds in terms of supervised learning objectives that measurably vary with different amounts of pruning.

Algorithm 1 A template for learning a partial policy $\psi = (\psi_0, \dots, \psi_{D-1})$. The template is instantiated by specifying the pairs of distributions and cost functions (μ_d, C_d) for $d \in \{0, \dots, D-1\}$. LEARN is an i.i.d. supervised learning algorithm that aims to minimize the expected cost of each ψ_d relative to C_d and μ_d .

```

1: procedure PARTIALPOLICYLEARNER( $\{(\mu_d, C_d)\}$ )
2:   for  $d = 0, 1, \dots, D-1$  do
3:     Sample a training set of states  $S_d$  from  $\mu_d$ 
4:      $\psi_d \leftarrow \text{LEARN}(S_d, C_d)$ 
5:   end for
6:   return  $\psi = (\psi_0, \psi_1, \dots, \psi_{D-1})$ 
7: end procedure

```

3 LEARNING PARTIAL POLICIES

Given μ_0 and D , we now develop reduction-style algorithms for learning partial policies. The algorithms reduce partial policy learning to a sequence of D i.i.d. supervised learning problems, each producing one partial policy component ψ_d . The supervised learning problem for ψ_d will be characterized by a pair (μ_d, C_d) , where μ_d is a distribution over states, and C_d is a cost function that, for any state s and action subset $A' \subseteq A$ assigns a prediction cost $C_d(s, A')$. The cost function is intended to measure the quality of A' with respect to including actions that are high quality for s . Typically the cost function is monotonically decreasing in A' and $C_d(s, A) = 0$.

In this section we assume the availability of an i.i.d. supervised learner LEARN that takes as input a set of states drawn from μ_d , along with C_d , and returns a partial policy component ψ_d that is intended to minimize the expected cost of ψ_d on μ_d , i.e. minimize $E[C_d(s, \psi_d(s))]$ for $s \sim \mu_d$. In practice, the specific learner will depend on the cost function and we describe our particular implementations in Section 4. Rather, in this section, we focus on defining reductions that allow us to bound the expected regret of ψ by the expected costs of the ψ_d returned by LEARN. The generic template for our reduction algorithms is shown in Algorithm 1.

In the following, our state distributions will be specified in terms of distributions induced by (complete) policies. In particular, given a policy π , we let $\mu_d(\pi)$ denote the state distribution induced by drawing an initial state from μ_0 and then executing π for d steps. Since we have assumed an MDP model or simulator, it is straightforward to sample from $\mu_d(\pi)$ for any provided π . Before proceeding we state a simple proposition that will be used to prove our regret bounds.

Proposition 3.1. *If a complete policy π is subsumed by partial policy ψ , then for any initial state distribution μ_0 , $\text{REG}(\mu_0, \psi) \leq E[V_0^*(s_0)] - E[V_0^\pi(s_0)]$, for $s_0 \sim \mu_0$.*

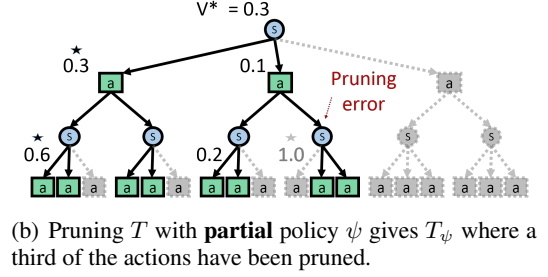
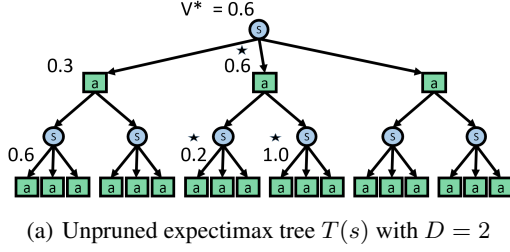


Figure 1: Unpruned and pruned expectimax trees with depth $D = 2$ for an MDP with $|A| = 3$ and two possible next states.

Proof. Since π is subsumed by ψ , we know that $Q_0^\psi(s, G_\psi(s)) = V_0^\psi(s) \geq V_0^\pi(s)$. Since for any a , $Q_0^*(s, a) \geq Q_0^\psi(s, a)$, we have for any state s , $Q_0^*(s, G_\psi(s)) \geq V_0^\pi(s)$. The result follows by negating each side of the inequality, followed by adding $V_0^*(s)$, and taking expectations. \square

Thus, we can bound the regret of a learned ψ if we can guarantee that it subsumes a policy whose expected value has bounded sub-optimality. Next we present three reductions for doing this, each having different requirements and assumptions.

3.1 OPI : OPTIMAL PARTIAL IMITATION

Perhaps the most straightforward idea for learning a partial policy is to attempt to find a partial policy that is usually consistent with trajectories of the optimal policy π^* . That is, each ψ_d should be learned so as to maximize the probability of containing actions selected by π_d^* with respect to the optimal state distribution $\mu_d(\pi^*)$. This approach is followed by our first algorithm called Optimal Partial Imitation (OPI). In particular, Algorithm 1 is instantiated with $\mu_d = \mu_d(\pi^*)$ (noting that $\mu_0(\pi^*)$ is equal to μ_0 as specified by the learning problem) and C_d equal to zero-one cost. Here $C_d(s, A') = 0$ if $\pi_d^*(s) \in A'$ and $C_d(s, A') = 1$ otherwise. Note that the expected cost of ψ_d in this case is equal to the probability that ψ_d does not contain the optimal action, which we will denote by $e_d^*(\psi) = \Pr_{s \sim \mu_d(\pi^*)}(\pi_d^*(s) \notin \psi_d(s))$.

A naive implementation of OPI is straightforward. We can generate length D trajectories by drawing an initial state from μ_0 and then selecting actions (approximately) according to π_d^* using standard unpruned search. Defined like this, OPI has the nice property that it only requires the ability to reliably compute actions of π_d^* , rather than requiring that we also estimate action values accurately. This allows us to exploit the fact that search algorithms such as UCT often quickly identify optimal actions, or sets of near-optimal actions, well before the action values have converged.

Intuitively we should expect that if the expected cost e_d^* is

small for all d , then the regret of ψ should be bounded, since the pruned search trees will generally contain optimal actions for state nodes. The following clarifies this dependence. For the proof, given a partial policy ψ , it is useful to define a corresponding complete policy ψ^+ such that $\psi_d^+(s) = \pi_d^*(s)$ whenever $\pi_d^*(s) \in \psi_d(s)$ and otherwise $\psi_d^+(s)$ is the lexicographically least action in $\psi_d(s)$. Note that ψ^+ is subsumed by ψ .

Theorem 3.2. *For any initial state distribution μ_0 and partial policy ψ , if for each $d \in \{0, \dots, D-1\}$, $e_d^*(\psi) \leq \epsilon$, then $\text{REG}(\mu_0, \psi) \leq \epsilon D^2$.*

Proof. Given the assumption that $e_d^*(\psi) \leq \epsilon$ and that ψ^+ selects the optimal action whenever ψ contains it, we know that $e_d^*(\psi^+) \leq \epsilon$ for each $d \in \{0, \dots, D\}$. Given this constraint on ψ^+ we can apply Lemma 3¹ from (Syed and Schapire, 2010) which implies $E[V_0^{\pi^+}(s_0)] \geq E[V_0^*(s_0)] - \epsilon D^2$, where $s_0 \sim \mu_0$. The result follows by combining this with Prop. 3.1. \square

This result mirrors work on reducing imitation learning to supervised classification (Ross and Bagnell, 2010; Syed and Schapire, 2010), showing the same dependence on the planning horizon. While space precludes details, it is straightforward to construct an example problem where the above regret bound is shown to be tight. This result motivates a learning approach where we have LEARN attempt to return ψ_d that each maximizes pruning (returns small action sets) while maintaining a small expected cost.

3.2 FT-OPI : FORWARD TRAINING OPI

OPI has a potential weakness, similar in nature to issues identified in prior work on imitation learning (Ross and Bagnell, 2010; Ross et al., 2011). In short, OPI does not train ψ to recover from its own pruning mistakes. Consider a node n in the optimal subtree of a tree $T(s_0)$ and suppose

¹The main result of (Syed and Schapire, 2010) holds for stochastic policies and requires a more complicated analysis that results in a looser bound. Lemma 3 is strong enough for deterministic policies.

that the learned ψ erroneously prunes the optimal child action of n . This means that the optimal subtree under n will be pruned from $T_\psi(s)$, increasing the potential regret. Ideally, we would like the pruned search in $T_\psi(s)$ to recover from the error gracefully and return an answer based on the best remaining subtree under n . Unfortunately, the distribution used to train ψ by OPI was not necessarily representative of this alternate subtree under n , since it was not an optimal subtree of $T(s)$. Thus, no guarantees about the pruning accuracy of ψ can be made under n .

In imitation learning, this type of problem has been dealt with via “forward training” of non-stationary policies (Ross et al., 2011) and a similar idea works for us. The Forward Training OPI (FT-OPI) algorithm differs from OPI only in the state distributions used for training. The key idea is to learn the partial policy components ψ_d in sequential order from $d = 0$ to $d = D - 1$ and then training ψ_d on a distribution induced by $\psi_{0:d-1} = (\psi_0, \dots, \psi_{d-1})$, which will account for pruning errors made by $\psi_{0:d-1}$. Specifically, recall that for a partial policy ψ , we defined ψ^+ to be a complete policy that selects the optimal action if it is consistent with ψ and otherwise the lexicographically least action. The state distributions used to instantiate FT-OPI is $\mu_d = \mu_d(\psi_{0:d-1}^+)$ and the cost function remains zero-one cost as for OPI. We will denote the expected cost of ψ_d in this case by $e_d^+(\psi) = \Pr_{s \sim \mu_d(\psi_{0:d-1}^+)}(\pi_d^*(s) \notin \psi_d(s))$, which gives the probability of pruning the optimal action with respect to the state distribution of $\psi_{0:d-1}^+$.

Note that as for OPI, we only require the ability to compute π^* in order to sample from $\mu_d(\psi_{0:d-1}^+)$. In particular, note that when learning ψ_d , we have $\psi_{0:d-1}$ available. Hence, we can sample a state for μ_d by executing a trajectory of $\psi_{0:d-1}^+$. Actions for ψ_d^+ can be selected by first computing π_d^* and selecting it if it is in ψ_d and otherwise selecting the lexicographically least action.

As shown for the forward training algorithm for imitation learning (Ross et al., 2011), we give below an improved regret bound for FT-OPI under an assumption on the maximum sub-optimality of any action. The intuition is that if it is possible to discover high-quality subtrees, even under sub-optimal action choices, then FT-OPI can learn on those trees and recover from errors.

Theorem 3.3. *Assume that for any state s , depth d , and action a , we have $V_d^*(s) - Q_d^*(s, a) \leq \Delta$. For any initial state distribution μ_0 and partial policy ψ , if for each $d \in \{0, \dots, D - 1\}$, $e_d^+(\psi) \leq \epsilon$, then $\text{REG}(\mu_0, \psi) \leq \epsilon \Delta D$.*

Proof. The theorem’s assumptions imply that ψ^+ and the search tree $T(s)$ satisfies the conditions of Theorem 2.2 of (Ross et al., 2011). Thus we can infer that $E[V_0^+(s_0)] \geq E[V_0^*(s_0)] - \epsilon \Delta D$, where $s_0 \sim \mu_0$. The result follows by combining this with Proposition 3.1. \square

Thus, when Δ is significantly smaller than D , FT-OPI has the potential to outperform OPI given the same bound on zero-one cost. In the worst case $\Delta = D$ and the bound will equal to that of OPI.

3.3 FT-QCM: FORWARD TRAINING Q-COST MINIMIZATION

While FT-OPI addressed one potential problem with OPI, they are both based on zero-one cost, which raises other potential issues. The primary weakness of using zero-one cost is its inability to distinguish between highly sub-optimal and slightly sub-optimal pruning mistakes. It was for this reason that FT-OPI required the assumption that all action values had sub-optimality bounded by Δ . However, in many problems, including those in our experiments, that assumption is unrealistic, since there can be many highly sub-optimal actions (e.g. ones that result in losing a game). This motivates using a cost function that is sensitive to the sub-optimality of pruning decisions.

In addition, it can often be difficult to learn a ψ that achieves both, a small zero-one cost and also provides significant pruning. For example, in many domains, in some states there will often be many near-optimal actions that are difficult to distinguish from the slightly better optimal action. In such cases, achieving low zero-one cost may require producing large action sets. However, learning a ψ that provides significant pruning while reliably retaining at least one near-optimal action may be easily accomplished. This again motivates using a cost function that is sensitive to the sub-optimality of pruning decisions, which is accomplished via our third algorithm, Forward Training Q-Cost Minimization (FT-QCM)

The cost function of FT-QCM is the minimum sub-optimality, or Q-cost, over unpruned actions. In particular, we use $C_d(s, A') = V_d^*(s) - \max_{a \in A'} Q_d^*(s, a)$. Our state distribution will be defined similarly to that of FT-OPI, only we will use a different reference policy. Given a partial policy ψ , define a new complete policy $\psi^* = (\psi_0^*, \dots, \psi_{D-1}^*)$ where $\psi_d^*(s) = \arg \max_{a \in \psi_d(s)} Q_d^*(s, a)$, so that ψ^* always selects the best unpruned action. We define the state distributions for FT-QCM as the state distribution induced by ψ^* , i.e. $\mu_d = \mu_d(\psi_{0:d-1}^*)$. We will denote the expected Q-cost of ψ at depth d to be $\Delta_d(\psi) = E[V_d^*(s_d) - \max_{a \in \psi_d(s_d)} Q_d^*(s_d, a)]$, where $s_d \sim \mu_d(\psi_{0:d-1}^*)$.

Unlike OPI and FT-OPI, this algorithm requires the ability to estimate action values of sub-optimal actions in order to sample from μ_d . That is, sampling from μ_d requires generating trajectories of ψ_d^* , which means we must be able to accurately detect the action in $\psi_d(s)$ that has maximum value, even if it is a sub-optimal action. The additional overhead for doing this during training depends on the search algorithm being used. For many algorithms,

near-optimal actions will tend to receive more attention than clearly sub-optimal actions. In those cases, as long as $\psi_d(s)$ includes reasonably good actions, there may be little additional regret. The following regret bound motivates the FT-QCM algorithm.

Theorem 3.4. *For any initial state distribution μ_0 and partial policy ψ , if for each $d \in \{0, \dots, D-1\}$, $\Delta_d(\psi) \leq \Delta$, then $\text{REG}(\mu_0, \psi) \leq 2D^{\frac{3}{2}}\sqrt{\Delta}$.*

Proof. Consider any pair of non-negative real numbers (ϵ, δ) such that for any d , the probability is no more than ϵ of drawing a state from $\mu_d(\psi^*)$ with Q-cost (wrt ψ) greater than δ . That is $\Pr(C_d(s_d, \psi_d(s_d)) \geq \delta) \leq \epsilon$. We will first bound $\text{REG}(\mu_0, \psi)$ in terms of ϵ and δ .

Let Π_δ be the set of policies for which all selected actions have regret bounded by δ . It can be shown by induction on the depth that for any $\pi \in \Pi_\delta$ and any state s , $V_0^\pi(s) \geq V_0^*(s) - \delta D$. For the chosen pair (ϵ, δ) we have that for a random trajectory t of ψ^* starting in a state drawn from μ_0 there is at most an ϵD probability that the Q-cost of ψ^* on any state of t is greater than δ . Thus, compared to a policy that always has Q-cost bounded by δ , the expected reduction in total reward of ψ for initial state distribution μ_0 is no more than ϵD^2 . This shows that

$$\begin{aligned} E[V_0^{\psi^*}(s_0)] &\geq \min_{\pi \in \Pi_\delta} E[V_0^\pi(s_0)] - \epsilon D^2 \\ &\geq E[V_0^*(s_0)] - \delta D - \epsilon D^2 \end{aligned}$$

Since ψ^* is subsumed by ψ , Proposition 3.1 implies that $\text{REG}(\mu_0, \psi) \leq \delta D + \epsilon D^2$.

We now reformulate the above bound in terms of the bound on expected Q-cost Δ assumed by the theorem. Since Q-costs are non-negative, we can apply the Markov inequality to conclude that $\Pr(C_d(s_d, \psi_d(s_d)) \geq \delta) \leq \frac{\Delta_d(\psi)}{\delta} \leq \frac{\Delta}{\delta}$. The pair $(\frac{\Delta}{\delta}, \delta)$ then satisfies the above condition for ψ^* . Thus, we get $\text{REG}(\mu_0, \psi) \leq \delta D + \frac{\Delta}{\delta} D^2$. The bound is minimized at $\delta = \sqrt{D\Delta}$, which yields the result. \square

FT-QCM tries to minimize this regret bound by minimizing $\Delta_d(\psi)$ via supervised learning at each step. Importantly, as we will show in our experiments, it is often possible to maintain small expected Q-cost with significant pruning, while the same amount of pruning would result in a much larger zero-one cost. It is an open problem as to whether this bound is tight in the worst case.

4 IMPLEMENTATION DETAILS

In this section, we describe our concrete implementation of the above abstract algorithms using the MCTS algorithm UCT (Kocsis and Szepesvári, 2006) for tree search.

UCT is a popular MCTS algorithm, perhaps most famous for leading to significant advances in computer Go (Gelly et al., 2006; Gelly and Silver, 2007). Space precludes a

complete description and we only outline the high-level ideas and issues relevant to our work. Given a root state and a time budget, UCT executes a series of Monte-Carlo simulations in order to build an asymmetric search tree that explores more promising parts of the tree first, compared to less promising parts. The key idea behind UCT is to utilize a variant of the multi-armed bandit algorithm UCB (Auer et al., 2002) for selecting actions during the Monte-Carlo simulations. The objective is to balance the exploration of less frequently visited parts of the tree versus exploring nodes that look most promising. State and action values for nodes in the UCT tree are estimated based on the average reward of Monte-Carlo trajectories that go through the nodes. In the limit, UCT will converge to optimal state and action values. In practice, the values computed in a UCT tree after a finite time budget are most accurate for higher quality actions, since they tend to be explored more often. It is straightforward to integrate a partial policy into UCT, by simply removing pruned actions from consideration.

Partial Policy Representation. Our partial policies operate by simply ranking the actions at a state and then pruning a percentage of the bottom actions. Specifically, partial policy components have the form $\psi_d(s \mid w_d, \sigma_d)$, parameterized by an n -dimensional weight vector w_d and pruning fraction $\sigma_d \in [0, 1]$. Given a state s , each action a is ranked according to the linear ranking function $f_d(s, a) = w_d^T \phi(s, a)$, where $\phi(s, a)$ is a user provided n -dimensional feature vector that describes salient properties of state-action pairs. Using f_d we can define a total order over actions, breaking ties lexicographically. $\psi_d(s \mid w_d, \sigma_d)$ is then equal to the set of the $\lceil (1 - \sigma_d)|A| \rceil$ highest ranked actions. This representation is useful as it allows us to separate the training of f_d from the selection of the pruning fraction.

Generating Training States. Each of our algorithms requires sampling training states from trajectories of particular policies that either require approximately computing π_d^* (OPI and FT-OPI) or also action values for sub-optimal actions (FT-QCM). Our implementation of this is to first generate a set of trees using substantial search, which provides us with the required policy or action values and then to sample trajectories from those trees.

More specifically, our learning algorithm is provided with a set of root states S_0 sampled from the target root distribution μ_0 . For each $s_0 \in S_0$ we run UCT for a specified time bound, which is intended to be significantly longer than the eventual online time bound. Note that the resulting trees will typically have a large number of nodes on the tree fringe that have been explored very little and hence will not be useful for learning. Because of this we select a depth bound D for training such that nodes at that depth or less have been sufficiently explored and have meaningful action values. The trees are then pruned until depth D .

Given this set of trees we can then generate trajectories using the MDP simulator of the policies specified for each algorithm. For example, OPI simply requires running trajectories through the trees based on selecting actions according to the optimal action estimates. The state at depth d along each trajectory is added to the data set for training ψ_d . FT-QCM samples states for training ψ_d by generating length d trajectories of $\psi_{0:d-1}^*$. Each such action selection requires referring to the estimated action values and returning the best one that is not pruned. The final state on the trajectory is then added to the training set for ψ_d . Note that since our approach generates multiple trajectories from each tree the training sets for each ψ_d are not truly i.i.d. This is an acceptable trade-off in practice since tree construction is expensive and this approach allows for many examples to be generated per tree.

Supervised Learner. It remains to specify how we implement the LEARN procedure. Each training set will consist of pairs $\{(s_i, c_i)\}$ where s_i is a state and c_i is a vector that assigns a cost to each action. For OPI and FT-OPI the cost vector assigns 0 to the optimal action and 1 to all other actions. For FT-QCM the c_i give the Q-costs of each action. We learn the partial policy by first learning the ranking function in a way that attempts to rank low-cost actions as highly as possible and then select an appropriate pruning percentage based on the learned ranker.

For rank learning, we follow a common approach of converting the problem to cost-sensitive binary classification. In particular, for a given example (s, c) we create a cost-sensitive classification example for each pair of actions a_1 and a_2 of the form $(\phi(s, a_1) - \phi(s, a_2), c(a_2) - c(a_1))$. Learning a linear classifier for such an example will attempt to rank a_1 above or below a_2 according to the cost difference. We apply an existing cost-sensitive learner (Langford, 2011) to learn a weight vector based on the pairwise data. Note that for zero-one loss, we do not create pairwise examples involving just sub-optimal actions since their cost difference will always be zero.

Finally, after learning the ranking function for a particular ψ_d , we must select an appropriate pruning percentage σ_d . In practice, we do this by analyzing the expected cost of ψ_d for a range of pruning values and select a pruning value that yields reasonably small costs. Section 6 gives details of the selections for our experiments.

5 RELATED WORK

While there is a large body of work on integrating learning and planning, to the best of our knowledge, we do not know of any work on learning partial policies for speeding up online MDP planning.

There are a number of efforts that study model-based reinforcement learning (RL) for large MDPs that utilize tree

search methods for planning with the learned model, e.g. RL using FSSS (Walsh et al., 2010), Monte-Carlo AIXI (Veness et al., 2011), and TEXPLORE (Hester and Stone, 2013). However, these methods focus on model/simulator learning and do not attempt to learn to speedup tree search using the learned models, which is the focus of our work.

A more related body of work is on learning search control knowledge in deterministic planning and games. One thread of work has been on learning knowledge for STRIPS-style deterministic planners, e.g. learning heuristics and policies for guiding best-first search (Yoon et al., 2008) or state ranking functions (Xu et al., 2009). The problem of learning improved leaf evaluation heuristics has also been studied in the context of deterministic real-time heuristic search (Bulitko and Lee, 2006). As one more example, evaluation functions have been learned for game tree search based on learning from “principle variations” of deep searches (Veness et al., 2009). The training data for this approach is similar in spirit to that of our OPI algorithm. Since these algorithms have been developed for deterministic settings, it is not straightforward to adapt them to the general MDP setting. Further, none of these existing methods, to our knowledge, have provided a theoretical analysis of the possible regret of using the learned knowledge, which is one of our main contributions.

There have been a number of efforts for utilizing domain-specific knowledge in order to improve/speedup MCTS, many of which are covered in a recent survey (Browne et al., 2012). A popular technique is to use a bias term $f(s, a)$ for guiding action selection during search. f is hand-provided in (Chaslot et al., 2007; Couëtoux et al., 2011) and learned in (Gelly and Silver, 2007; Sorg et al., 2011). Generally there are a number of parameters that dictate how strongly $f(s, a)$ influences search and how that influence decays as search progresses. In our experience, tuning the parameters for a particular problem can be tedious and difficult to do in a domain-independent way. Similar issues hold for the approach in (Gelly and Silver, 2007) which attempts to learn an approximation of Q^* and then initializes search nodes with the estimate. In (Sorg et al., 2011), control knowledge is learned via policy-gradient techniques in the form of a reward function and used to guide MCTS with the intention of better performance given a time budget. So far, however, the approach has not been analyzed formally and has not been demonstrated on large MDPs. Experiments in small MDPs have also not demonstrated improvement in terms of wall clock time over vanilla MCTS.

Finally, MCTS methods often utilize hand-coded or learned “default policies” (e.g., MoGo (Gelly et al., 2006)) to improve anytime performance. While this has shown some promise in specific domains such as Go, where the policies can be highly engineered for efficiency, we have found that the computational overhead of using learned default poli-

cies is often too high a price to pay. In particular, most such learned policies require the evaluation of a feature vector at each state encountered, or for each state-action pair. This may cause orders of magnitude fewer trajectories to be executed compared to vanilla MCTS. In our experience, this can easily lead to degraded performance per unit time. Furthermore, there is little formal understanding about how to learn such rollout policies in principled ways, with straightforward approaches often yielding decreased performance (Silver and Tesauro, 2009).

6 EXPERIMENTS

We consider learning partial policies in two domains.

Yahtzee is a classic dice game with ≈ 17 actions on average. Actions correspond to selecting subsets of dice to roll and selecting categories to score. The objective is to maximize the sum of category scores. We use 26 features for learning the partial policy which encode the impact of a roll action or select action for each of the 13 categories.

Galcon is a two-player real-time strategy game with approximately 174 actions per move on average (max. ≈ 800). The objective is to maximize one’s own population on a 2-D grid of planets by directing variable-sized fleets of units between planets to capture enemy planets or fortify one’s own. Transitions are stochastic when opposing populations battle over a planet. We use 20 real-valued state-action features that coarsely encode the impact of an action on population size, planet ownership, etc.

In what follows, we experiment with the two extreme algorithms, OPI and FT-QCM. The intermediate algorithm FT-OPI is excluded in this paper due to the time required to perform large-scale evaluations across a wide spectrum of time bounds. Preliminary experiments suggest that FT-OPI performs between OPI and FT-QCM.

Setup. For training we provide our learning algorithms with root states generated by playing 100 full games allowing UCT 64 seconds per move per tree. This produces thousands of root states along with the search trees, which we use for training as described in Section 4. Due to the large action and stochastic branching factors, even with 64 seconds per move, we selected $D = 3$ for learning since the value estimates produced by UCT for deeper nodes were not accurate. Note that in our evaluations we do not limit the rollouts of UCT to depth 3, rather the rollouts proceed until terminal states, which provide a longer term heuristic evaluation for tree nodes.² In order to select the pruning fractions σ_d , in results not shown for space reasons, we plotted the supervised losses achieved for various learned ranking functions for a variety of σ_d . We found that in

²When UCT generates tree nodes at depths greater than D , in these experiments we prune using ψ_D . However, we have confirmed experimentally that typically such nodes are not visited frequently enough for pruning to have a significant impact.

Galcon a value of $\sigma_d = 0.75$ provided a good trade-off point since the supervised average costs began to sharply increase beyond that point. For Yahtzee we found that $\sigma_d = 0.75$ was a good trade-off point for all depths except the root. At the root we used a less aggressive pruning fraction $\sigma_0 = 0.5$ since the cost increased more quickly with σ_0 . Finally, all of our online evaluations of the learned partial policies within UCT are averaged across 1000 games and 95% confidence intervals are plotted.

FT-QCM v OPI. Figures 2(a) and 2(d) compare the search performance of UCT given various amounts of time per move, when using the partial policies learned by FT-QCM to that learned by OPI. FT-QCM has clear wins across most of the anytime curve, with OPI achieving parity at sufficiently large budgets in both domains. In results not shown for space reasons, we repeated this experiment for a variety of sets of the pruning fraction σ_d . In all cases, OPI never outperforms FT-QCM for any setting of σ_d and at best, achieves parity. Based on this result the remainder of our experimental evaluation is based on FT-QCM.

FT-QCM v baselines. We now compare the performance of pruning with FT-QCM partial policies to other approaches for injecting control knowledge into UCT. Most other approaches require an action scoring function of some form, for which we use the ranking function f_0 learned for the root partial policy by FT-QCM. The first baseline is *vanilla* UCT without any knowledge injection. Second, we use (decaying) heuristic bias (*HB*) in UCT as discussed in the related work, using the best settings found by experimenting with different methods of setting the bias component. Our third baseline uses a softmax policy, parameterized by f_0 , as the default policy (*DP*). During a rollout, an action is sampled with probability $\exp(f_0(s, a)) / \sum_{a' \in A(s)} \exp(f_0(s, a'))$ instead of uniform random selection. Our fourth and last baseline, *greedy*, does not search and selects root actions greedily according to the learned root ranking function. Note that this final baseline can be viewed as an attempt to apply traditional imitation learning to our problem.

Figures 2(b) and 2(e) give the results for Galcon and Yahtzee. For Galcon, which is two-player, the curves correspond to each method playing against *vanilla* UCT using the same budget on wall-clock time. The heuristic bias (*HB*) technique is unable to show any improvement over *vanilla* when evaluated in terms of performance versus wall-clock time. This is likely due to f_0 being inaccurate and highly biased when used as an evaluation function. Furthermore, tuning the complex interaction of Q estimates, exploration bonus and noisy bias terms is challenging and time-consuming. Note that *HB* does not decrease search performance unlike the informed default policy (*DP*) baseline. Here, the cost of using the expensive softmax rollout policy dwarfs any benefit. *DP* only seems to “pay for itself” at the largest budgets where parity is

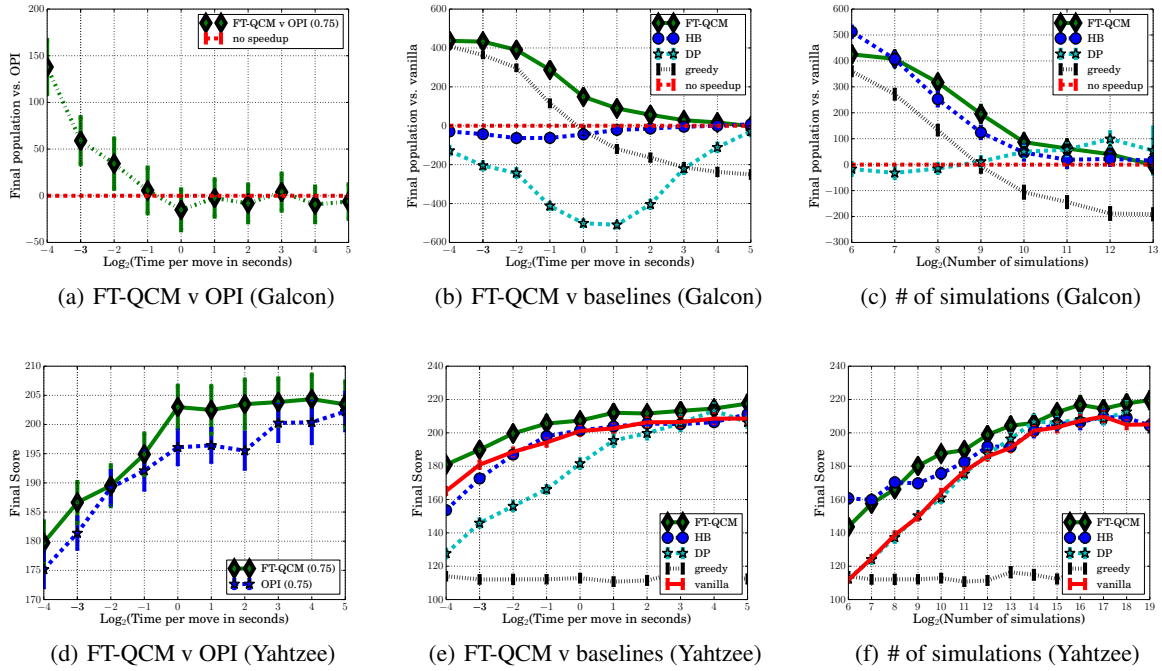


Figure 2: The first column shows a comparison between partial policies learned by FT-QCM and OPI when used inside UCT for different time bounds. In (a), the baseline player is OPI instead of *vanilla* (b,c). The second column shows the main experimental result where *FT-QCM* consistently outperforms baselines in both domains. *DP* and *HB* are surprisingly bad when knowledge costs are taken into account. The third column shows performance under zero knowledge costs by measuring the budget in simulation counts. Again, *FT-QCM* does well consistently but now *DP* and *HB* perform better.

achieved with the other baselines. The last baseline *greedy* demonstrates the limited strength of the reactive policy which is only competitive in Galcon when the search is very time-constrained. In Yahtzee, *greedy* is not competitive at any budget. We note that we have attempted to apply alternative imitation learning approaches to arrive at improved reactive policies, with little improvement. It appears that our feature representation is not powerful enough to accurately discriminate optimal actions. By relaxing our goal to learning a partial policy and integrating with search, we are able to benefit significantly from learning.

In both domains, FT-QCM outperforms every baseline at every budget considered. In Galcon, it has large wins on the left of the anytime curve and smaller wins on the right (not visible due to scale). In Yahtzee as well, FT-QCM achieves higher scores consistently. Furthermore, it continues to improve even at the largest budgets. In Yahtzee, increasing average score above 200 is extremely challenging and FT-QCM's performance improvement is significant.

The cost of knowledge. It is initially surprising that the heuristic bias and informed default policy methods do not improve over *vanilla* MCTS. It turns out these methods do well as long as the cost of using knowledge is set to zero. That is, budgets are specified in terms of the number of simulations conducted by UCT instead of wall-clock time.

The difference is shown in Figures 2(c) and 2(f) where all the planners were re-run using simulations instead of time. Now *HB* outperforms *vanilla* in both domains while *DP* is competitive or wins by small amounts. However, our method FT-QCM is again better across most of the anytime curve in both domains. This result shows the importance of evaluating in terms of wall-clock time, which has not always been common practice in prior work.

7 SUMMARY

We have shown algorithms for offline learning of partial policies for reducing the action branching factor in time-bounded tree search. The algorithms leverage a reduction to i.i.d supervised learning and are shown to have bounds on the worst case regret. Experiments on two challenging domains show significantly improved anytime performance in Monte-Carlo Tree Search. One line of future work involves more sophisticated uses of partial policies during search (e.g., progressive widening, time-aware search control) for further improvements in anytime performance. We are also interested in iterating the learning process, where we use a learned ψ to guide deeper search in order to generate training data for learning an even better ψ .

Acknowledgements

This work was supported by NSF grant IIS-1219258.

References

- Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *MLJ*, 47(2-3):235–256, May 2002.
- Andrew G. Barto, Steven J. Bradtke, and Satinder P. Singh. Learning to act using real-time dynamic programming. *AI*, 72(1-2):81–138, 1995.
- Blai Bonet and Hector Geffner. Action selection for MDPs: Anytime AO* versus UCT. In *AAAI*, 2012.
- Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of Monte Carlo tree search methods. *CIG*, 4(1):1–43, 2012.
- Vadim Bulitko and Greg Lee. Learning in real-time search: A unifying framework. *JAIR*, 25:119–157, 2006.
- Guillaume Chaslot, Mark Winands, Jaap H van den Herik, Jos Uiterwijk, and Bruno Bouzy. Progressive strategies for Monte-Carlo tree search. pages 655–661, 2007.
- Adrien Couëtoux, Jean-Baptiste Hoock, Nataliya Sokolovska, Olivier Teytaud, and Nicolas Bonnard. Continuous upper confidence trees. In *LION*, pages 433–445. Springer, 2011.
- Sylvain Gelly and David Silver. Combining online and offline knowledge in UCT. In *ICML*, pages 273–280. ACM, 2007.
- Sylvain Gelly, Yizao Wang, Rémi Munos, and Olivier Teytaud. Modification of UCT with patterns in Monte-Carlo Go. Technical report, INRIA, 2006.
- Todd Hester and Peter Stone. Texplora: real-time sample-efficient reinforcement learning for robots. *MLJ*, 90(3):385–429, 2013.
- Michael Kearns, Yishay Mansour, and Andrew Y Ng. A sparse sampling algorithm for near-optimal planning in large Markov decision processes. *MLJ*, 49(2-3):193–208, 2002.
- Levente Kocsis and Csaba Szepesvári. Bandit based Monte-Carlo planning. In *ECML*, pages 282–293. 2006.
- John Langford. Vowpal Wabbit. URL https://github.com/JohnLangford/vowpal_wabbit/wiki, 2011.
- Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *AISTATS*, pages 661–668, 2010.
- Stéphane Ross, Geoffrey J Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*, pages 627–635, 2011.
- David Silver and Gerald Tesauro. Monte-Carlo simulation balancing. In *ICML*, pages 945–952, 2009.
- Jonathan Sorg, Satinder P Singh, and Richard L Lewis. Optimal rewards versus leaf-evaluation heuristics in planning agents. In *AAAI*, 2011.
- Umar Syed and Robert E Schapire. A reduction from apprenticeship learning to classification. In *NIPS*, pages 2253–2261, 2010.
- Joel Veness, David Silver, Alan Blair, and William W Cohen. Bootstrapping from game tree search. In *NIPS*, pages 1937–1945, 2009.
- Joel Veness, Kee Siong Ng, Marcus Hutter, William Uther, and David Silver. A Monte-Carlo AIXI approximation. *JAIR*, 40(1):95–142, 2011.
- Thomas J. Walsh, Sergiu Goschin, and Michael L. Littman. Integrating sample-based planning and model-based reinforcement learning. In *AAAI*, 2010.
- Yuehua Xu, Alan Fern, and Sungwook Yoon. Learning linear ranking functions for beam search with application to planning. *JMLR*, 10:1571–1610, December 2009.
- Sungwook Yoon, Alan Fern, and Robert Givan. Learning control knowledge for forward search planning. *JMLR*, 9:683–718, 2008.

Estimating Accuracy from Unlabeled Data

Emmanouil Antonios Platanios

Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA 15213

e.a.platanios@cs.cmu.edu

Avrim Blum

Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213

avrim@cs.cmu.edu

Tom Mitchell

Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA 15213

tom.mitchell@cs.cmu.edu

Abstract

We consider the question of how unlabeled data can be used to estimate the true accuracy of learned classifiers. This is an important question for any autonomous learning system that must estimate its accuracy without supervision, and also when classifiers trained from one data distribution must be applied to a new distribution (e.g., document classifiers trained on one text corpus are to be applied to a second corpus). We first show how to estimate error rates exactly from unlabeled data when given a collection of competing classifiers that make independent errors, based on the agreement rates between subsets of these classifiers. We further show that *even when the competing classifiers do not make independent errors, both their accuracies and error dependencies can be estimated* by making certain relaxed assumptions. Experiments on two data real-world data sets produce estimates within a few percent of the true accuracy, using solely unlabeled data. These results are of practical significance in situations where labeled data is scarce and shed light on the more general question of how the consistency among multiple functions is related to their true accuracies.

1 INTRODUCTION

Estimating accuracy of classifiers is central to machine learning and many other fields. Traditionally, one estimates accuracy of a function based on its performance over a set of labeled test examples. This paper considers the question of under what conditions is it possible to estimate accuracy based instead on *unlabeled data*. We show that accuracy can be estimated exactly from unlabeled data in the case that at least three different approximations to the same function are available, so long as these functions make independent errors and have better than chance ac-

curacy. More interestingly, we show that even if one does not assume independent errors, one can still estimate accuracy given a sufficient number of competing approximations to the same function, by viewing the degree of independence of those approximations as an optimization criterion. We present experimental results demonstrating the success of this approach in estimating classification accuracies to within a few percentage points of their true value, in two diverse domains.

We consider a “multiple approximations” problem setting in which we have several different approximations, $\hat{f}_1, \dots, \hat{f}_N$, to some target boolean classification function, $f : \mathcal{X} \rightarrow \{0, 1\}$, and we wish to know the true accuracies of each of these different approximations, using only unlabeled data. The multiple functions can be from any source - learned or manually constructed. One example of this setting that we consider here is taken from the Never Ending Language Learning system (NELL) [Carlson et al., 2010]. NELL learns classifiers that map noun phrases (NPs) to boolean categories such as fruit, food and vehicle. For each such boolean classification function, NELL learns several different approximations based on different views of the NP. One approximation is based on the orthographic features of the NP (e.g., if the NP ends with the letter string “burgh”, it may be a city), whereas another uses phrases surrounding the NP (e.g., if the NP follows the word sequence “mayor of”, it may be a city). Our aim in this paper is to find a way to estimate the error rates of each of the competing approximations to f , using only unlabeled data (e.g., many unlabeled NPs in the case of NELL).

2 RELATED WORK

Other researchers have considered variants of this “multiple approximations” setting. For example, [Blum and Mitchell, 1998] introduced the co-training algorithm which uses unlabeled data to train competing approximations to a target function by forcing them to agree on classifications of unlabeled examples. Others have used the disagreement rate between competing approximations as a distance met-

ric to perform model selection and regularization [Schuurmans et al., 2006; Bengio and Chapados, 2003]. Balcan et al. [2013] used disagreement along with an ontology to estimate the error of the prediction vector for multi-class prediction, from unlabeled data, under an assumption of independence of the input features given the labeling. Parisi et al. [2014] proposed a spectral method used to rank classifiers based on accuracy and combine their outputs to produce one final label, also under an assumption of independence of the input features given the labeling. Moreover, there has been work at developing more robust semi-supervised learning algorithms by using the concept of agreement rates [Collins and Singer, 1999] or some task specific constraints [Chang et al., 2007] to decide what should be added to the training data set. However, very few have tried to directly estimate actual per function error rates using agreement rates. In [Dasgupta et al., 2001] the authors PAC-bound the error rates using the pairwise agreement rates only, under the assumption that the functions make independent errors, and [Madani et al., 2004] estimate the average error of two predictors using their disagreements. [Donmez et al., 2010] is one of the few to estimate per-function error rates from unlabeled data. Here, the authors estimate the prediction risk for each function under the assumption that the true probability distribution of the output labels is known. Much of the emphasis of their work is on methods that use the known label distribution to estimate the error rate even of a single classifier, but agreements are used as well, especially under the assumption of conditional independence. In contrast, we propose here several methods for estimating actual function error rates from agreement rates, without making these assumptions.

The main contributions of this paper include: (1) formulating the problem of estimating the error rate of each of several approximations to the same function, based on their agreement rates over *unlabeled data*, as an optimization problem, (2) providing two different analytical methods that estimate error rates from agreement rates in this setting, one based on a set of simultaneous equations relating accuracies, agreements, and error dependencies, and a second, based on maximizing data likelihood, and (3) demonstrating the success of these two methods in two very different real-world problems. We consider our proposed methods a first step towards developing a self-reflection framework for autonomous learning systems.

3 PROPOSED METHODS

We introduce two different methods to estimate the error rates of binary functions in the multiple approximations setting described in section 1. Both methods are based on the idea of looking at the consistency between the different functions' predictions in order to determine the error rates of those functions. The first method consists of matching

the sample agreement rates of the functions with the exact formulas of those agreement rates written in terms of the functions' error rates. For the second method, we formulate the functions' predictions consistency as a probabilistic model and solve for the maximum likelihood estimate (MLE) of their error rates. Both methods estimate the individual error rates for each function, as well as the joint error rates of all possible subsets of those functions, based on the predictions made by these functions over a sample of unlabeled instances X_1, \dots, X_S .

In the following sections we denote the input data by X and the true binary output label by Y . We assume the input data X are drawn from some unknown distribution $P(X) = \mathcal{D}$, and $Y \in \{0, 1\}$. Let us consider N functions, $\hat{f}_1(X), \dots, \hat{f}_N(X)$ which attempt to model the mapping from X to Y . For example, each function might be the result of a different learning algorithm, or might use a different subset of the features of X as input. We define the error event $E_{\mathcal{A}}$ of a set of functions \mathcal{A} as an event in which every function in \mathcal{A} makes an incorrect prediction:

$$E_{\mathcal{A}} = \bigcap_{i \in \mathcal{A}} [\hat{f}_i(X) \neq Y], \quad (1)$$

where \cap denotes the set intersection operator and where \mathcal{A} contains the indices of the functions. We define the error rate of a set of functions \mathcal{A} (i.e. the probability that all functions in \mathcal{A} make an error together) as:

$$e_{\mathcal{A}} = \mathbb{P}_{\mathcal{D}}(E_{\mathcal{A}}), \quad (2)$$

where $\mathbb{P}_{\mathcal{D}}(\cdot)$ denotes the probability of an event under the distribution over the input data X .

3.1 AGREEMENT RATES METHOD

Let us define the agreement rate $a_{\mathcal{A}}$, for a set of functions \mathcal{A} as the probability that all of the functions' outputs¹ are the same:

$$a_{\mathcal{A}} = \mathbb{P}_{\mathcal{D}}\left(\left\{\hat{f}_i(X) = \hat{f}_j(X), \forall i, j \in \mathcal{A} : i \neq j\right\}\right). \quad (3)$$

This quantity can be defined in terms of the error rates of the functions in \mathcal{A} . In order to understand how we can write the agreement rate in terms of error rates let us consider a simple example where $\mathcal{A} = \{i, j\}$ (i.e. consider just the pairwise agreement rate between the functions f_i and f_j). The probability of two functions agreeing is equal to the probability that both make an error, plus the probability that neither makes an error:

$$a_{\{i,j\}} = \mathbb{P}_{\mathcal{D}}(E_{\{i\}} \cap E_{\{j\}}) + \mathbb{P}_{\mathcal{D}}(\bar{E}_{\{i\}} \cap \bar{E}_{\{j\}}), \quad (4)$$

where $\bar{\cdot}$ denotes the complement of a set. By using De Morgan's laws and the inclusion-exclusion principle we obtain,

¹Here, "outputs" is equivalent to "predictions".

using the notation defined in equation (2), an expression for the agreement rate between the two functions, in terms of their individual error rates, and their joint error rate:

$$a_{\{i,j\}} = 1 - e_{\{i\}} - e_{\{j\}} + 2e_{\{i,j\}}. \quad (5)$$

In the same way we obtain the following general result for the agreement rate of a set of functions \mathcal{A} of arbitrary size:

$$\begin{aligned} a_{\mathcal{A}} &= \mathbb{P}_{\mathcal{D}}\left(\bigcap_{i \in \mathcal{A}} E_i\right) + \mathbb{P}_{\mathcal{D}}\left(\bigcap_{i \in \mathcal{A}} \bar{E}_i\right), \\ &= e_{\mathcal{A}} + 1 - \mathbb{P}_{\mathcal{D}}\left(\bigcup_{i \in \mathcal{A}} E_i\right), \\ &= e_{\mathcal{A}} + 1 + \sum_{k=1}^{|\mathcal{A}|} \left[(-1)^k \sum_{\substack{I \subseteq \mathcal{A} \\ |I|=k}} e_I \right], \end{aligned} \quad (6)$$

where \cup denotes the set union operator and $|\cdot|$ denotes the number of elements in a set. For the first line we used the fact that the two events, $\{\bigcap_{i \in \mathcal{A}} E_i\}$ and $\{\bigcap_{i \in \mathcal{A}} \bar{E}_i\}$, are mutually exclusive, for the second line we used one of De Morgan's laws and for the last line we used the inclusion-exclusion principle.

In the next section we examine the most basic case, assuming that functions make independent errors and have error rates below 0.5, showing that we can solve exactly for the error rates provided that we have at least 3 different functions. In the subsequent section we examine the most general case, assuming that we have N functions that make errors with unknown inter-dependencies, and show that we can formulate this as a constrained numerical optimization problem whose objective function reflects a soft prior assumption regarding the error dependencies. Experimental results presented in a later section demonstrate the practical utility of this approach, producing estimated error rates that are within a few percentage points of the true error rates, *using only unlabeled data*.

3.1.1 3 Functions That Make Independent Errors

When we have 3 functions that make independent errors we can replace the $e_{\{i,j\}}$ term in equation (5) with the term $e_{\{i\}}e_{\{j\}}$. In this case we have only 3 unknown variables (i.e. the individual function error rates) and we have $\binom{3}{2} = 3$ equations (i.e. equation (5), for $1 \leq i < j \leq 3$). Therefore, we can directly solve for each error rate in terms of the three observed agreement rates:

$$e_{\{i\}} = \frac{c \pm (1 - 2a_{\{j,k\}})}{\pm 2(1 - 2a_{\{j,k\}})}, \quad (7)$$

where $i \in \{1, 2, 3\}$, $j, k \in \{1, 2, 3\} \setminus i$ with $j < k$ and:

$$c = \sqrt{(2a_{\{1,2\}} - 1)(2a_{\{1,3\}} - 1)(2a_{\{2,3\}} - 1)}, \quad (8)$$

where, for a set B and an element of that set b , the notation $B \setminus b$ denotes the set containing all elements in B except b . In practical applications, we can estimate the agreement rates among the competing functions, using a sample of unlabeled data X_1, \dots, X_S , as follows:

$$\hat{a}_{\{i,j\}} = \frac{1}{S} \sum_{s=1}^S \mathbb{I}\{\hat{f}_i(X_s) = \hat{f}_j(X_s)\}, \quad (9)$$

where $\mathbb{I}\{\cdot\}$ evaluates to one if its argument statement is true and to zero otherwise.

In most practical applications the competing functions do not make independent errors. We next consider the more difficult problem of estimating the error rates from agreement rates, but without assuming independence of the function error events.

3.1.2 N Functions That Make Dependent Errors

When we have N functions that make dependent errors we rely on the agreement rate equation (6). We consider the agreement rates for all sets $\mathcal{A} = \{A \subseteq \{1, \dots, N\} : |A| \geq 2\}$ of functions (the agreement rate is uninformative for less

KEY IDEA

The significance of equations (5) and (6) is that they relate the different agreement rates $a_{\mathcal{A}}$, which are easily estimated from *unlabeled* data, to the true error rates $e_{\mathcal{A}}$ of the functions, which are difficult to estimate without labeled data. Note that if we have a system of such equations with rank equal to the number of error rates mentioned, then we can solve exactly for these error rates in terms of the observed agreement rates. This is not the case in general, because given a set of functions, $\hat{f}_1, \dots, \hat{f}_N$, we obtain $2^N - N - 1$ agreement rate equations (one for each subset of two or more functions) expressed in terms of $2^N - 1$ error rates (one for each non-empty subset of functions). However, if we assume that the errors made by the N individual functions are independent, then we can express all of the $2^N - 1$ error rates in terms of N single-function error rates (e.g., $e_{\{i,j\}} = e_{\{i\}}e_{\{j\}}$) and we can then solve exactly for all error rates (given the additional assumption that error rates are better than chance). Furthermore, if we are unwilling to make the strong assumption that errors of individual functions are independent, then we can instead solve for the set of error rates that minimize the dependence among errors (e.g., among the infinite solutions to the underdetermined set of equations, we choose the solution that minimizes $\sum_{i,j} (e_{\{i,j\}} - e_{\{i\}}e_{\{j\}})^2$ - this idea can be easily extended to larger subsets than simply pairs of functions). The key idea in this paper is that the correspondence between easily-observed agreement rates and hard-to-observe error rates given by these equations can be used as a practical basis for estimating true error rates from unlabeled data.

than two functions) and we obtain $2^N - N - 1$ equations by matching equation (6) to the sample agreement rate for each possible subset of functions. Given a sample of unlabeled data X_1, \dots, X_S , the sample agreement rate is defined as:

$$\hat{a}_{\mathcal{A}} = \frac{1}{S} \sum_{s=1}^S \mathbb{I} \left\{ \hat{f}_i(X_s) = \hat{f}_j(X_s), \forall i, j \in \mathcal{A} : i \neq j \right\}, \quad (10)$$

and is an unbiased estimate of the true agreement rate. Moreover, our unknown variables are all the individual function error rates along with all of the possible joint function error rates (let us denote the vector containing all those variables by e); that is a total of $2^N - 1$ unknown variables.

The set of $2^N - N - 1$ equations involving $2^N - 1$ unknown variables yields an underdetermined system of equations with an infinite number of possible solutions. We therefore cast this problem as a constrained optimization problem where the agreement equations form constraints that must be satisfied and where we seek the solution that minimizes the following objective:

$$c(e) = \sum_{\mathcal{A}: |\mathcal{A}| \geq 2} \left(e_{\mathcal{A}} - \prod_{i \in \mathcal{A}} e_i \right)^2. \quad (11)$$

It can be seen that we are basically trying to minimize the dependence between the error events², while satisfying all of the agreement rates constraints. We saw in section 3.1.1 that if we assume that the error events are independent, then we can obtain an exact solution. By defining our optimization problem in this way we are effectively relaxing this constraint by saying that we want to find the error rates that satisfy our constraints and that are, at the same time, as independent as possible. Most existing methods trying to estimate function error rates using only unlabeled data assume that the error events are independent; the main novelty of this method lies in the fact that *we relax all those assumptions* and make no hard or strict assumptions about our functions.

Note that we could also define different objective functions based on information we might have about our function approximations or based on different assumptions we might want to make. For example, one could try minimizing the sum of the squares of all the error rates (i.e. the L_2 norm of e) in order to obtain the most optimistic error rates that satisfy the agreement rates constraints. The novelty of our method partly lies in the formulation of the error rates estimation problem using only unlabeled data as a constrained optimization problem.

In this section we defined the model we are using for this method and the optimization problem we wish to solve. We call this method the AR method (i.e. Agreement Rates

²That can be seen from the fact that when the error events are independent we have that $e_{\mathcal{A}} = \prod_{i \in \mathcal{A}} e_i$.

method). In section 3.3 we define additional constraints that both this method and the maximum likelihood method (described in the next section) use.

3.2 MAXIMUM LIKELIHOOD METHOD

In this section we define a probabilistic model of the consistency in the functions' outputs, considering the most general case of having N functions that make potentially dependent errors. Let us denote the outputs of the functions on an i.i.d. sample of data X_1, \dots, X_S by $\hat{Y}_s = [\hat{f}_1(X_s), \dots, \hat{f}_N(X_s)]$, for $s \in \{1, \dots, S\}$. The \hat{Y}_s 's are independent and therefore we can define the likelihood of our model as:

$$L(e) = \mathbb{P}_{\mathcal{D}}(\hat{Y}_1, \dots, \hat{Y}_S | e) = \prod_{s=1}^S \mathbb{P}_{\mathcal{D}}(\hat{Y}_s | e), \quad (12)$$

where the parameter vector e contains all of the possible error events probabilities. More specifically, it contains all the e_I , for all $I \subseteq \{1, \dots, N\}$ and $|I| \in \{1, \dots, N\}$.

Now, \hat{Y}_s contains all the function outputs given data sample X_s . In order to compute $\mathbb{P}_{\mathcal{D}}(\hat{Y}_s | e)$ we need to consider the following two cases:

1. All functions agree with each other (i.e. \hat{Y}_s is a vector of all 1's or all 0's).
2. The functions can be split into two non-empty groups: those that output 1 and those that output 0.

The groups of functions with the same output can also be viewed as maximal cliques in the graph whose nodes consist of the functions and whose edges consist of the agreements between the functions (i.e. when there is an agreement between two functions there is an edge connecting their corresponding nodes in the graph and when there is no agreement between them there is no edge). By using this representation we call the first case the "one clique case" and the second case the "two cliques case". We are now going to consider those two cases separately.

One Clique Case: Let us denote the set of all function indices in the clique by \mathcal{C} (i.e. $\mathcal{C} = \{1, \dots, N\}$). In this case, either all functions make an error or none of them does. Therefore, for the probability of the current sample we have that:

$$\begin{aligned} \mathbb{P}_{\mathcal{D}}(\hat{Y}_s | e) &= \mathbb{P}_{\mathcal{D}}\left(\bigcap_{i \in \mathcal{C}} E_i\right) + \mathbb{P}_{\mathcal{D}}\left(\bigcap_{i \in \mathcal{C}} \bar{E}_i\right), \\ &= e_{\mathcal{C}} + 1 - \mathbb{P}_{\mathcal{D}}\left(\bigcup_{i \in \mathcal{C}} E_i\right), \\ &= e_{\mathcal{C}} + 1 + \sum_{k=1}^{|\mathcal{C}|} \left[(-1)^k \sum_{\substack{I \subseteq \mathcal{C} \\ |I|=k}} e_I \right], \end{aligned} \quad (13)$$

following an equivalent derivation to the one we used when defining the agreement rates in section 3.1.

Two Cliques Case: Let us denote the set of function indices in the first clique by \mathcal{C}_1 and those in the second clique by \mathcal{C}_2 . Then, we have two possible events:

1. All functions in \mathcal{C}_1 make an error and none of the functions in \mathcal{C}_2 makes an error.
2. All functions in \mathcal{C}_2 make an error and none of the functions in \mathcal{C}_1 makes an error.

Let $\mathbb{P}_{\mathcal{D}}^1(\hat{\mathbf{Y}}_s|e)$ denote the probability of $\hat{\mathbf{Y}}_s$ given that the first event of those two occurs, and let $\mathbb{P}_{\mathcal{D}}^2(\hat{\mathbf{Y}}_s|e)$ denote the probability of $\hat{\mathbf{Y}}_s$ given that the second event of those two occurs. It can be easily seen that those two events are mutually exclusive and so we have that $\mathbb{P}_{\mathcal{D}}(\hat{\mathbf{Y}}_s|e) = \mathbb{P}_{\mathcal{D}}^1(\hat{\mathbf{Y}}_s|e) + \mathbb{P}_{\mathcal{D}}^2(\hat{\mathbf{Y}}_s|e)$.

Following an equivalent derivation to the one we used when defining the agreement rates in section 3.1 we have that:

$$\begin{aligned} \mathbb{P}_{\mathcal{D}}^1(\hat{\mathbf{Y}}_s|e) &= \mathbb{P}_{\mathcal{D}}\left(\left[\bigcap_{i \in \mathcal{C}_1} E_i\right] \cap \left[\bigcap_{j \in \mathcal{C}_2} \bar{E}_j\right]\right), \\ &= \mathbb{P}_{\mathcal{D}}\left(\left[\bigcap_{i \in \mathcal{C}_1} E_i\right] \cap \overline{\left[\bigcup_{j \in \mathcal{C}_2} E_j\right]}\right), \quad (14) \\ &= e_{\mathcal{C}_1} + \sum_{k=1}^{|\mathcal{C}_2|} \left[(-1)^k \sum_{\substack{I \subseteq \mathcal{C}_2 \\ |I|=k}} e_{\{I \cup \mathcal{C}_1\}}\right]. \end{aligned}$$

For the second line we used one of De Morgan's laws and for the last line we used a modified form of the inclusion-exclusion principle. To understand the step we used to obtain the last line in the above equation let us consider a simple case with example events A , B_1 and B_2 . It is clear from the Venn diagram in figure 1, on the right, that:

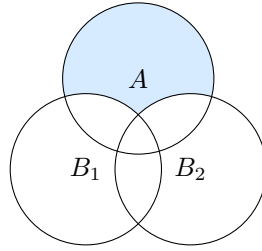


Figure 1: Venn diagram for the simple example used to explain our last step in equation (14).

$$\begin{aligned} \mathbb{P}(A \cap \overline{[B_1 \cup B_2]}) &= \mathbb{P}(A) \\ &\quad - \mathbb{P}(A \cap B_1) - \mathbb{P}(A \cap B_2) \quad (15) \\ &\quad + \mathbb{P}(A \cap B_1 \cap B_2). \end{aligned}$$

Our last step in equation (14) follows from extending this result using the inclusion-exclusion principle. In the same way we also get that:

$$\mathbb{P}_{\mathcal{D}}^2(\hat{\mathbf{Y}}_s|e) = e_{\mathcal{C}_2} + \sum_{k=1}^{|\mathcal{C}_1|} \left[(-1)^k \sum_{\substack{I \subseteq \mathcal{C}_1 \\ |I|=k}} e_{\{I \cup \mathcal{C}_2\}}\right]. \quad (16)$$

Having defined our likelihood function all that remains is to describe the optimization problem that we are solving to obtain the maximum likelihood estimate for e . We use the negative of the natural logarithm of the likelihood function (i.e. the log-likelihood function) as our objective function, which we want to minimize, and the details of how we perform the optimization are described in section 3.3. We call this the MLE method.

3.2.1 Regularization

In this subsection we define a method which is a slight modification of the above MLE method, in that it uses a modified objective function. The objective function considered in our MLE method is non-convex and hence has many local maxima. In order to avoid getting stuck into one of those local maxima, or at least try to avoid it, we here add a regularization term to the objective function. Following the same argument we used in constructing the objective function of the AR method, we define our new objective function, which we wish to minimize, as:

$$c(e) = -\log L(e) + \lambda \sum_{\mathcal{A}: |\mathcal{A}| \geq 2} \left(e_{\mathcal{A}} - \prod_{i \in \mathcal{A}} e_i\right)^2, \quad (17)$$

where λ is a hyperparameter whose value can be chosen arbitrarily. We call this the maximum a posteriori (MAP) method because the added term is equivalent to adding a Gaussian prior of a special form to the error rates estimates³. As we will see in the experiments section the performance of this method will depend on the value chosen for λ .

3.3 OPTIMIZATION

In sections 3.1 and 3.2 we defined the optimization problems corresponding to each of our methods. For all methods we use the TOMLAB Base Module v.7.7 “conSolve” solver. In the following sections we discuss: (1) some additional constraints that apply to all methods, (2) extensions of our approach to the case where multiple approximations are learned for each of several different target functions, and (3) an approximation that can make our methods much faster, more scalable and maybe even more accurate.

3.3.1 Error Rates Constraints

Our unknown variables include both individual function error rates and joint function error rates of those events. We need to impose constraints on the values that the joint function error rates can take. These constraints follow from basic rules of probability and set theory; they represent bounding joint event probabilities using the corresponding marginal event probabilities. These constraints are defined

³ λ can be interpreted as a function of the variance of that prior.

by the following equation:

$$e_{\mathcal{A}} \leq \min_{i \in \mathcal{A}} e_{\mathcal{A} \setminus i}, \quad (18)$$

for $|\mathcal{A}| \geq 2$. Furthermore, regarding the individual function error rates, it is easy to see that if we transform all e_i , for $i = 1, \dots, N$, to $1 - e_i$, the resulting agreement rates are equal to the original ones. A similar result holds for the likelihood function. In order to make our models identifiable we add the constraint that $e_i \in [0, 0.5]$, for $i = 1, \dots, N$, which simply means that our functions/binary classifiers perform better than chance. It is thus a very reasonable constraint⁴.

3.3.2 Dealing With Multiple Classification Problems

Up to this point we have assumed that there is a single target function and multiple approximations to that function. More generally though, we might have multiple target functions, or problem settings, and a common set of learning algorithms used for learning each one of those. For example, this is the case in NELL, where the different target functions correspond to different boolean classification problems (e.g., classifying NPs as “cities” or not, as “locations” or not, etc.). Multiple learning methods are utilized to approximate each one of those target functions (e.g., a classifier based on the NP orthography, a second classifier based on the NP contexts, etc.), so that each such classification problem, or target function, corresponds to an instance of our “multiple approximations” problem setting.

Of course we can apply our AR or our MLE methods to estimate accuracies separately for each target classification problem (and that is what we actually did in our experiments described in section 4). However, when we have multiple target functions to be learned and multiple learning methods shared across each, there is an interesting opportunity to further couple the error estimates across these different target functions. In equations (11) and (17) we introduced terms to minimize the dependency between the error rates of competing approximations. In the case where we have multiple target functions, we might introduce additional terms to capture other relevant assumptions. For example, we could introduce a term to minimize the difference in error dependencies between two learning methods across multiple classification problems (e.g., we could choose to minimize the difference in error dependencies between orthography-based and context-based classifiers trained for different classification problems).

3.3.3 Approximating High Order Error Rates

Once the agreement rate estimates (number of occurrences of each clique formation in the case of the MLE method

⁴It is important to understand here that in order for our methods to work in the first place, this constraint *must* hold for the classifiers that we are considering.

and the MAP method) have been calculated, the execution time of the optimization procedure for all proposed methods does not depend on the number of provided data samples⁵, S . It does however depend on the number of functions, N . This can be easily seen by considering the number of unknown variables we have which is equal to $2^N - 1$. As will be shown in section 4, the performance of all methods, in terms how good the obtained function error rate estimates are, increases with an increasing number of functions, N . It is therefore not a good idea to try to reduce N . So, we instead propose a way to reduce the execution time of the optimization procedure by approximating high order error rates, instead of estimating them directly.

We can estimate high order joint function error rates⁶ using lower order function error rates by using the following formula, for $|\mathcal{A}| > M_e$, where M_e is chosen arbitrarily:

$$e_{\mathcal{A}} = \frac{1}{|\mathcal{A}|} \sum_{i \in \mathcal{A}} e_{\mathcal{A} \setminus i} e_i. \quad (19)$$

With a high value of M_e we obtain better estimates but execution time is larger, and vice-versa. This estimate is based on the fact that the higher the order of the function error rates, the less significant the impact of an independence assumption between them.

Furthermore, the only available information regarding high order error rates comes from high order sample agreement rates⁶, $\hat{a}_{\mathcal{A}}$, which will likely be very noisy estimates of the true agreement rates. That is because there will be very few data samples where all of the functions in \mathcal{A} will agree and therefore the sample agreement rate will be computed using only a small number of data samples resulting in a noisy estimate of the true agreement rate. This motivates not directly estimating high order error rates, but instead approximating them using low order error rates. In fact, in the case that the sample agreement rates are too noisy, this approximation might even increase the quality of the obtained error rate estimates. By approximating high order error rates in the way described earlier, we are effectively ignoring the corresponding high order sample agreement rates (i.e. they are not used in our estimation) for the AR method.

⁵Even the execution time of the optimization procedure for the MLE method, which seems to depend on S , does not actually depend on it because there is only a fixed number of possible clique combinations one can obtain for a given number of functions, N . That number is equal to $2^N - 1$. In a large data sample we will have a lot of repeated samples in terms of the maximal cliques that they result in. We can compute the log-likelihood term for each one of those cliques only once and multiply it by the number of samples in which they each appear. This way our algorithm’s execution time only depends on N .

⁶By “order” of an error rate, $e_{\mathcal{A}}$, or agreement rate, $a_{\mathcal{A}}$, we mean the number of functions in set \mathcal{A} , or simply $|\mathcal{A}|$.

4 EXPERIMENTS

We present here experiments using two very different data sets, to explore the ability of our methods to estimate error rates in realistic settings without domain-specific tuning. For both data sets we used a set of labeled data examples to perform our experiments. We used the data samples without their corresponding labels to estimate agreement rates, and to then estimate error rates using our methods. We used the same examples with their labels to estimate each function’s true error rate, which we call from here on the “true error rate” of the function.

NELL Data Set: This data set consists of data samples where we use four binary logistic regression (LR) classifiers to predict whether a NP belongs to a specific category in the NELL knowledge base (e.g. is Monongahela a river?). The domain in this case is defined by the category (e.g. “beverage” and “river” are two different domains) and the four classifiers used were the following: (1) **ADJ**: A LR classifier that uses as features the adjectives that occur with the NP over millions of web pages, (2) **CMC**: A LR classifier that considers orthographic features of the NP (e.g. does the NP end with the letter string “burgh”? - more details can be found in [Carlson et al., 2010]), (3) **CPL**: A LR classifier that uses as features words and phrases that appear with the NP, and (4) **VERB**: A LR classifier that uses as features verbs that appear with the NP. Table 1 lists the NELL categories that we used as the domains in our experiments, along with the number of labeled examples available per category. Note the NP features used by these four classifiers are somewhat independent given the correct classification label.

Brain Data Set: Functional Magnetic Resonance Imaging (fMRI) data were collected while 8 subjects read a chapter from a popular novel [Rowling, 2012], one word at a time. The classification task is to find which of two 40 second long story passages correspond to an unlabeled 40 second time series of fMRI neural activity. For this binary classification task, we consider eight different classifiers, each making its prediction based on a different representation of the text passage (e.g., the number of letters in each word of the text passage, versus the part of speech of each word, versus emotions experienced by characters in the story, etc.). In this case different domains correspond to 11 different locations in the brain and we have 924 labeled examples per location. Additional details can be found in [Wehbe et al., 2014].

Our experimental results for both data sets are presented and discussed in the following two sections. As a performance measure we use the mean absolute deviation (MAD) between the true function error rates and the function error rates estimated from unlabeled data (i.e. we sum the absolute values of the element-wise differences of the true error

Table 1: A listing of the 15 NELL categories we used as the domains in our experiments, along with the number of labeled examples available per category.

Category	# Examples	Category	# Examples
animal	20,733	food	19,566
beverage	18,932	fruit	18,911
bird	19,263	muscle	21,606
bodypart	21,840	person	21,700
city	21,778	protein	21,811
disease	21,827	river	21,723
drug	20,452	vegetable	18,826
fish	19,162		

rates vector and the estimated error rates vector). We compute the MAD for the individual function error rates alone, for the pairwise function error rates (i.e. for $|\mathcal{A}| = 2$) alone and for all function error rates together. Note the higher order error rates are quite small, because it is rare for every one of the competing functions to simultaneously err. Therefore, we consider the individual and pairwise function error rates to be most diagnostic of how well our approach is working.

4.1 NELL DATA SET RESULTS

We initially applied the AR method using only the ADJ, the CPL and the VERB classifiers, while assuming that they make independent errors. The method for estimating error rates in this case is described in section 3.1.1. In this case we estimate only the individual function error rates. The resulting MAD is 2.82×10^{-2} ; that is, the average error estimate is within a few percent of the true error. Although encouraging, this MAD is poor in comparison to our less restricted methods described below, and indicates that the assumption that the classifiers make independent errors is an incorrect in this case (and in most other cases as a matter of fact). Some of the obtained error rates are not even within the interval $[0, 0.5]$ and are thus obviously incorrect, since we know by the construction of the problem that the true error rates lie in this interval. From now on we consider only the more general case of N functions that make dependent errors, thus making no independence assumptions.

Table 2 presents results for all three of our methods used with the entire NELL data set. It includes the results obtained when using all available data samples (i.e. the numbers shown in table 1) and when using only 50 data samples per category. It is clear from this table that the more data samples we have the better our methods perform, presumably due to the the more accurate estimates of the true agreement rates for the AR method, and for the other two methods, to the larger volume of evidence we have to incorporate into our likelihood. Furthermore, we see that the AR method performs significantly better than the other two methods. This could possibly be attributed to the fact that

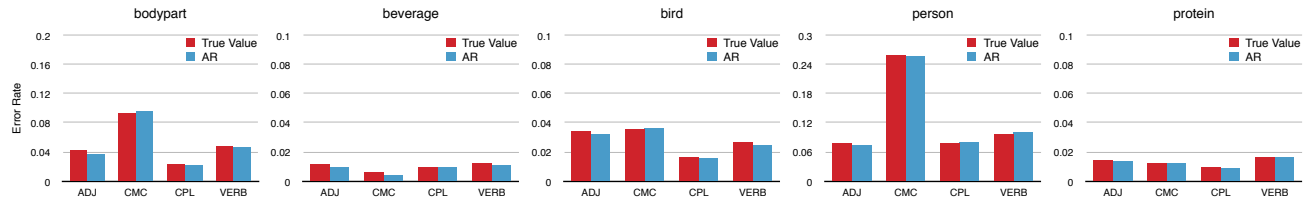


Figure 2: True errors (red bars) versus errors estimated from unlabeled data using AR method (blue bars), for four competing function approximations (ADJ, CMC, CPL and VERB), to five different target function domains (i.e. “bodypart”, “beverage”, “bird”, “person” and “protein”) using the NELL data set. Note each plot uses a different vertical scale to make it easier to observe the accuracy of the error rates estimates.

Table 2: Mean absolute deviation (MAD) of individual (Ind.), pairwise (Pair.) and all function error rates for the NELL data set, for all three proposed methods and for the cases where we use all of the available data samples and only 50 data samples per domain.

$\times 10^{-2}$	All Data Samples			50 Data Samples		
	Ind.	Pair.	All	Ind.	Pair.	All
AR	0.49	0.31	0.29	0.82	0.39	0.40
MLE	2.77	2.19	1.84	20.06	19.96	15.42
MAP	1.54	1.30	1.08	13.11	15.17	11.14

for this method we solve a convex optimization problem, whereas for the other two we solve a non-convex one and we possibly get stuck in local minima. Better numerical optimization solvers could possibly help with that. Finally, the MAP method performs better than the MLE method, presumably reflecting the correctness of our prior which attempts to minimize dependencies in errors across competing approximations. We did discover that the performance of the MAP method depends strongly on the choice of the λ parameter. In this case we selected $\lambda = 10$ simply because that value gave the regularization term the same order of magnitude as the log-likelihood term in the objective function. Moreover, now it becomes clear why the 2.82×10^{-2} MAD that we obtained when we assumed independent error events is quite a bad result. The AR method manages to achieve an MAD that is almost 6 times better than that.

We also run an experiment by using the approximation described in section 3.3.3 and setting $M_e = 2$ (i.e. considering only pairwise agreement rates). The individual functions MAD in this case was 0.52×10^{-2} , the pairwise one was 0.35×10^{-2} and the overall one was 0.31×10^{-2} . These results are worse than the ones we obtained without using this approximation, as expected, but they are still very good. This is important because it shows that this proposed approximation method is useful (there was a significant speedup as well - the code run 3 times faster).

From these results it is clear that the AR method, which also happens to be the simplest and fastest of the three methods we propose, performs better than the other proposed methods, for this data set, and also does not require tuning any parameters (as opposed to the MAP method).

INDEPENDENCE ASSUMPTION WEAKNESS

In order to make it more clear that the independence assumption is not very appropriate even in the case of NELL where a significant amount of effort has been put into having the NELL classifiers make independent errors, we provide here a measure of that dependence. We compute the following quantity for each domain:

$$\frac{1}{Z} \sum_{i,j} \left| \frac{e_{\{i,j\}}}{e_{\{i\}}e_{\{j\}}} - 1 \right|, \quad (20)$$

where Z is the total number of terms in the sum, and we average over all domains. That gives us a measure of the average dependence of the functions error rates across all domains. If the functions make independent errors, then this quantity should be equal to 0. We computed this quantity for the NELL data set using the sample error rates, which are an estimate of the true error rates (a pretty accurate estimate since we have about 20,000 data samples per domain), and we obtained a value of 8.1770, which is indeed quite far from 0. That indicates why our methods, and especially the AR method, do so much better than the exact solution when assuming independent errors.

Figure 2 provides a plot of the estimated error rates for the AR method, along with the true error rates for five randomly selected NELL classification problems (plots for all regions are not included in this paper due to space constraints). This plot gives an idea of how good the AR estimates are, and helps to make sense of the reported MAD values. As is easily seen in this plot, irrespective of the exact error estimate *the ranking of the competing function approximations based on error rate is recovered exactly* by using the AR method. And that is in fact true for each of the 15 NELL target function classification problems we evaluated – not only for the five shown in this figure.

4.2 BRAIN DATA SET RESULTS

Table 3 presents results for the brain data set, obtained when using 4 of the 8 competing function approximations (randomly selected to be classifiers 1, 3, 4 and 5) and when using all 8 of them. It is clear from that table that the more competing classifiers used, the better the quality of the resulting estimates. When using all 8 classifiers the AR method performs significantly better than the other two

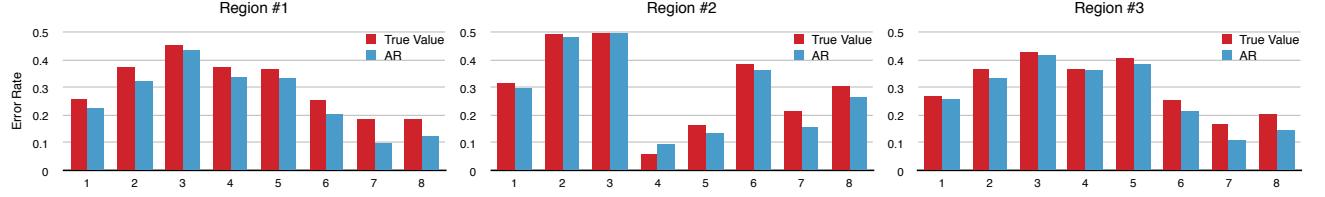


Figure 3: True errors (red bars) versus errors estimated from unlabeled data using AR method (blue bars), for eight competing function approximations (based on different story features), to three different target function domains (using neural activity from three different brain regions) using the brain data set. Note estimates from unlabeled data are quite close to true errors.

Table 3: Mean absolute error (MAE) of individual (Ind.), pairwise (Pair.) and all function error rates for the brain data set, for all three proposed methods and for the cases where we use 4 classifiers and 8 classifiers.

$\times 10^{-2}$	4 Classifiers			8 Classifiers		
	Ind.	Pair.	All	Ind.	Pair.	All
AR	10.97	6.60	6.50	4.36	4.14	2.01
MLE	10.60	8.34	7.64	32.02	12.33	4.50
MAP	9.61	18.19	11.16	27.95	18.60	7.26

methods. We have also included a plot of the estimated error rates for the AR method, along with the true error rates, for three randomly selected brain regions (i.e. domains), in figure 3 (it is clear from the figure that we can recover the ranking of the classifiers based on error rate, using the AR method, for this data set as well).

Note for this data set, for the case when we use 8 classifiers, the MLE method and the MAP method both perform poorly. This can probably be attributed to the optimization algorithm not being able to deal with those problems very well, due to their high dimensionality and non-convexity. These results could probably be improved by choosing a different optimization algorithm better suited for those problems. It is interesting to note that when we use only 4 classifiers the MLE and the MAP methods perform slightly better than the AR method in estimating the individual function error rates. However, they perform significantly worse when dealing with higher order error rates and so, overall, the AR method still dominates. Note that for this data set, for the MAP method, we also selected $\lambda = 10$ for the same reasons as for the NELL data set.

We also ran an experiment using the approximation described in section 3.3.3 and setting $M_e = 2$ (i.e. considering only pairwise agreement rates). The individual functions MAD in this case was 4.40×10^{-2} , the pairwise one was 4.06×10^{-2} and the overall one was 1.90×10^{-2} . These results are slightly better than the ones we obtained without using this approximation. This is important because it shows once again that this proposed approximation method is useful (there was a significant speedup as well - the code run 8 times faster). The better accuracy could possibly be attributed to two factors: (i) the problem is of much lower dimensionality and so the optimization algorithm might be

dealing better with it and (ii) the high order sample agreement rates might have been bad estimates of the true agreement rates due to insufficient data and so they might have affected our methods negatively.

5 CONCLUSION

We have introduced the concept of estimating the error rate of each of several approximations to the same function, based on their agreement rates over *unlabeled data* and we have provided three different analytical methods to do so: the AR method, the MLE method and the MAP method. Our experiments showed that the AR method performs significantly better than the other two methods for both data sets we considered. Our results are very encouraging and suggest that function agreement rates are indeed very useful in estimating function error rates. We consider this work to be a first step towards developing a *self-reflection framework* for autonomous learning systems.

There are several directions we would like to pursue to further improve upon the methods introduced here. Firstly, we wish to explore other interesting natural objectives one can aim to optimize, as described in section 3.1.2. It would also be very interesting to explore possible generalizations of our models to non-boolean, discrete-valued functions, or even to real-valued functions. Finally, apart from simply estimating function error rates, we want to explore how the obtained error rate estimates can be used to improve the learning ability of a system such as NELL, for example. In this context, we could try using our estimates in order to develop a more robust co-training framework. One very direct application of our methods would be to use the estimated error rates and their dependencies in order to combine the functions' outputs and obtain one final output.

Acknowledgements

We thank Leila Wehbe for providing us with the brain data set and Alan Ritter and Siddharth Varia for providing us with the NELL data set. Finally, we thank the previously mentioned people, Jayant Krishnamurthy and the anonymous reviewers for their helpful comments. This research has been supported in part by DARPA under contract number FA8750-13-2-0005 and in part by NSF grants IIS-1065251 and CCF-1116892.

References

- Maria-Florina Balcan, Avrim Blum, and Yishay Mansour. Exploiting Ontology Structures and Unlabeled Data for Learning. *International Conference on Machine Learning*, pages 1112–1120, 2013.
- Yoshua Bengio and Nicolas Chapados. Extensions to Metric-Based Model Selection. *Journal of Machine Learning Research*, 3:1209–1227, March 2003.
- Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory, COLT’ 98*, pages 92–100, 1998. doi: 10.1145/279943.279962.
- Andrew Carlson, Burr Settles, Justin Betteridge, Bryan Kisiel, Estevam R Hruschka Jr, and Tom M Mitchell. Toward an Architecture for Never-Ending Language Learning. In *Conference on Artificial Intelligence (AAAI)*, pages 1–8, 2010.
- Ming-Wei Chang, Lev Ratinov, and Dan Roth. Guiding Semi-Supervision with Constraint-Driven Learning. In *Annual Meeting of the Association of Computational Linguistics*, pages 280–287, Prague, Czech Republic, June 2007.
- Michael Collins and Yoram Singer. Unsupervised Models for Named Entity Classification. In *Joint Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 1–11, 1999.
- Sanjoy Dasgupta, Michael L Littman, and David McAllester. PAC Generalization Bounds for Co-training. In *Neural Information Processing Systems*, pages 375–382, 2001.
- Pinar Donmez, Guy Lebanon, and Krishnakumar Balasubramanian. Unsupervised Supervised Learning I: Estimating Classification and Regression Errors without Labels. *Journal of Machine Learning Research*, 11:1323–1351, April 2010.
- Omid Madani, David M Pennock, and Gary W Flake. Co-Validation: Using Model Disagreement on Unlabeled Data to Validate Classification Algorithms. In *Neural Information Processing Systems*, pages 1–8, 2004.
- Fabio Parisi, Francesco Strino, Boaz Nadler, and Yuval Kluger. Ranking and combining multiple predictors without labeled data. *Proceedings of the National Academy of Sciences*, pages 1–28, January 2014.
- J.K. Rowling. *Harry Potter and the Sorcerer’s Stone*. Harry Potter US. Pottermore Limited, 2012. ISBN 9781781100271.
- Dale Schuurmans, Finnegan Southey, Dana Wilkinson, and Yuhong Guo. Metric-Based Approaches for Semi-Supervised Regression and Classification. In *Semi-Supervised Learning*, pages 1–31. 2006.
- Leila Wehbe, Brian Murphy, Partha Talukdar, Alona Fyshe, Aaditya Ramdas, and Tom Mitchell. Predicting brain activity during story processing. *in review*, 2014.

k -NN Regression on Functional Data with Incomplete Observations

Sashank J. Reddi
Machine Learning Department
Carnegie Mellon University
sjakkamr@cs.cmu.edu

Barnabás Póczos
Machine Learning Department
Carnegie Mellon University
bapoczos@cs.cmu.edu

Abstract

In this paper we study a general version of regression where each covariate itself is a functional data such as distributions or functions. In real applications, however, typically we do not have direct access to such data; instead only some noisy estimates of the true covariate functions/distributions are available to us. For example, when each covariate is a distribution, then we might not be able to directly observe these distributions, but it can be assumed that i.i.d. sample sets from these distributions are available. In this paper we present a general framework and a k -NN based estimator for this regression problem. We prove consistency of the estimator and derive its convergence rates. We further show that the proposed estimator can adapt to the local intrinsic dimension in our case and provide a simple approach for choosing k . Finally, we illustrate the applicability of our framework with numerical experiments.

1 INTRODUCTION

Machine learning has undergone a paradigm shift in the recent times. Traditional machine learning techniques focused on *simple* form of data such as features modeled as vectors in \mathbb{R}^p . However, with the advent of modern data collection methods datasets have not only become huge but also more complex, often involving objects like distributions, functions, and sets. Consider the example of brain connectivity mapping data. The brain contains billions of neurons with several trillion physical connections. Neuroimaging approaches like Diffusion Spectrum Imaging (DSI) attempt to visualize the underlying anatomical architecture of neural pathways by creating 3D probability distributions of water diffusion along nerve fiber bundles. In this ex-

ample the input data consists of distributions, instead of simple finite dimensional vectors. Likewise, there are many instances where the training data consists of functions. For example, whenever we encounter with time series data (e.g. time series of commodity's price, patient's health monitor, energy usage data), then we can always think of the instances as functions whose domain is the time.

Unfortunately, our understanding of algorithms for such complex data is still limited. Most of the existing machine learning and statistical techniques cannot handle such data, often resorting to ad-hoc approaches; thereby ignoring the underlying rich structure in the data. This necessitates the development of a different machine learning paradigm where the *true structure* in the complex data can be exploited. The goal of this paper is to further advance our knowledge of such algorithms.

One of the central issues working with complex functional data is that it is typically difficult to obtain the exact data (functions or distributions). Hence, our access to the data is often restricted to some noisy estimate of the data. For example, when the input variables are distributions, then it is more natural to assume that we only have finite samples from the distributions, but the true distributions (such as their pdf or cdf) are unknown to us. The empirical distribution can be viewed as a noisy estimate of the distribution. Similarly, in the case of function regression, we have the function values at some selected points rather than whole the function itself. We use the terms “measurement error” and “error in variable” to emphasize this issue of noise in the data.

Although, there have been a few attempts to tackle the issues of “error in variables” [3], most of the earlier works do not fully exploit the scenario where we have control over the measurement error. This is particularly relevant to the applications we are interested in such as distribution regression, where we can obtain more accurate measurements of the data by obtaining

more samples from the distribution.

While working with complex data, it is often desirable to have a simple yet powerful algorithms. One such algorithm often used in traditional machine learning is the k -Nearest Neighbor (k -NN) regression estimation. k -NN based algorithms are easy to use and robust. Furthermore, thanks to the extensive research on nearest neighbor search, there are many efficient algorithms for finding the nearest neighbors [1, 22, 5]. Additionally, k -NN estimators have the virtue of adapting to the local structure of the data [10]. Due to these factors, k -NN estimators are well-suited for complex data. However, very little is understood about these estimators in the context of functional data. To this end, we study the problem of k -NN regression on functional data with measurement error. We present our results in a rather broad framework since working within a general framework allows us to use the same tools across different settings and understand the underlying principles of k -NN estimators.

Main Contributions: Our contributions can be summarized as follows: (i) We provide a general framework for analysis of k -NN estimators for functional data. (ii) We prove consistency of the estimators under weak assumptions. (iii) We derive convergence rates for the estimators. (iv) We provide probabilistic bounds which exploit the local intrinsic structure of the probability measure. (v) We provide an adaptive procedure to select k by exploiting the local intrinsic structure. (vi) We apply the framework in two interesting settings, namely distribution regression and function regression. Due to space constraints, we relegate few longer proofs to the appendix.

2 RELATED WORK

Our work is related to functional data analysis, a new exciting field of statistics. We refer interested readers to [4, 17] for a comprehensive treatment of the topic. However, note that most of these works assume direct access to the covariates without any measurement error. This does not fit our framework for regression over distributions or functions.

One popular approach to deal with *distribution* covariates in ML tasks is to first embed the distributions into a reproducing kernel Hilbert space (RKHS) and then solve the learning problem using the standard machinery of kernel methods [20, 6, 18]. There are both parametric and non-parametric methods proposed along these lines. Parametric methods usually fit a parametric model to distributions for estimating inner products [9, 8, 12]. Few non-parametric methods for distributions also exist. For example, set kernels (since the samples from the distributions are repre-

sented by sets) or kernels over distributions may be used. In this context, it is worthwhile to note that the representer theorem was recently generalized for the space of probability distributions [13].

More recently, Póczos et al. [16] proposed a kernel regression approach for solving the regression problem with distribution covariates and real-valued responses. Convergence and sample complexity of the estimator were analyzed in the paper. Oliva et al. [14] provided a similar analysis for the case where the response is also a distribution. Function regression has also gained considerable interest recently. Oliva et al. [15] provided a functional analogue to the LASSO and studied the statistical properties of the estimator. The functional output case has been studied in [11]. None of these works, however, provide an adaptive algorithm which exploits the local structure of the data, such as the local intrinsic dimensionality. This is an important issue, because this dimension plays an important role in the convergence rate. To design efficient algorithms, it is important to be able to adapt to the local intrinsic dimensions. Another important difference between these algorithms and the estimator we propose here is that none of these algorithms are based on k -NN.

Our work is also related to the error in variables model [3]. However, unlike the latter case where the error is $O(1)$ and is not decreasing, we have control over the error and hence, can obtain very accurate (but expensive) measurements. This is true in the applications of our interest like distribution and function regression. As we will see later, we can exploit this additional flexibility to obtain faster rates of convergence.

There has been fairly extensive research on k -NN estimators for regression problem. Kpotufe et al. [10] study k -NN regression and show that it adapts to local intrinsic dimension. Furthermore, they also provide a simple method to choose k that nearly achieves the minimax rate. But these works do not address the problem of our concern, namely k -NN estimators for functional data with error in measurement.

Notation: The symbol $\mathbb{P}(E)$ is used to denote the probability of event E . We use $X \sim \mathbb{P}$ to denote that the random variable X has probability distribution \mathbb{P} . We use the $[n]$ and $i : j$ to denote the set $\{1, \dots, n\}$ and $\{i, \dots, j\}$ respectively. The symbol $\mathbb{E}(X)$ is used to denote the expectation of random variable X . We use $B(P, r)$ to denote a ball of radius r centered around P (where P is a point in some metric space).

3 PRELIMINARIES

We start this section with a formal discussion of k -NN based *regression* estimators. We denote by (\mathcal{P}, ρ) a

metric space \mathcal{P} with distance measure ρ . We assume that the space with this metric ρ is bounded, i.e., there exists a $\nu > 0$ such that $\rho(P, Q) \leq \nu$ for all $P, Q \in \mathcal{P}$. In a typical regression setting, we have m i.i.d samples $(\mathbf{P}, \mathbf{Y}) = \{(P_i, Y_i)\}_{i=1}^m$ from some unknown distribution over $(\mathcal{P} \times \mathbb{R})$, which is the space of input-output pairs. For example, many machine learning applications usually deal with finite dimensional Euclidean spaces, i.e., $\mathcal{P} = \mathbb{R}^p$ and ρ is the Euclidean distance. We assume that for our observations $\{Y_i\}$ it holds that

$$Y_i = f(P_i) + \gamma_i, \quad i \in [m],$$

where f is a regression function $f : \mathcal{P} \rightarrow \mathbb{R}$, and γ_i 's are noise variables with $\mathbb{E}[\gamma_i] = 0$ and variance $\mathbb{E}[\gamma_i^2] = \sigma^2$. We assume that the functional f is L -Lipschitz, i.e., $|f(P) - f(P')| \leq L\rho(P, P')$ for all $P, P' \in \mathcal{P}$. We use μ and μ_m to denote the marginal distribution and the empirical distribution on \mathcal{P} respectively. k -NN based regression is fairly well-understood when $\mathcal{P} = \mathbb{R}^p$ and ρ is the Euclidean distance [7].

In functional data analysis, as mentioned earlier, it is usually not possible to obtain the samples P_i exactly, and hence we have to deal with a *noisy* representation of P_i . Our goal, however, is still the same as in standard regression problems: to recover the function f . The type of representation generally depends on the application. For example, in the case of distribution regression (i.e., \mathcal{P} is the space of continuous distributions), we only have access to the samples from the distributions (and not the distributions themselves).

To formalize the notion of noisy representation of the input data, assume that we have m i.i.d. samples $(\hat{\mathbf{P}}, \mathbf{Y}) = \{(\hat{P}_i, Y_i)\}_{i=1}^m$ instead of (\mathbf{P}, \mathbf{Y}) . Here \hat{P}_i denotes the empirical estimation of distribution P_i .

In what follows, we will discuss the details of k -NN regression for functional data. We first look at the case of fixed k (given as an input). We will later investigate an approach to adaptively select k . The regression estimate at P (function or distribution) using $(\hat{\mathbf{P}}, \mathbf{Y})$ is defined as follows:

$$\hat{f}(P, \hat{P}_1, \dots, \hat{P}_m) = \sum_{j=1}^m Y_j W_j(P, \hat{P}_1, \dots, \hat{P}_m), \quad (1)$$

where

$$W_j(P, \hat{P}_1, \dots, \hat{P}_m) = \begin{cases} \frac{1}{k} & \text{if } \hat{P}_j \text{ is } k\text{-NN of } P \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

For the sake of brevity, we use $\hat{f}(P)$ and W_j to denote $\hat{f}(P, \hat{P}_1, \dots, \hat{P}_m)$ and $W_j(P, \hat{P}_1, \dots, \hat{P}_m)$ respectively. It should also be noted that more general approaches, such as the generalized version of k -NN with non-uniform weights can also be used through the means

of a kernel function [10]. For simplicity we only analyze the case of uniform weight in this paper, but all of our results can be extended to the aforementioned scenario.

Before we delve into the technical details of the regression estimator in Equation (1), we have to introduce the definition of doubling dimension, which will help us deriving upper bounds on the generalization error of the estimator. We just briefly describe the definition here, and refer interested readers to [10] for more details.

Definition 1. (*Doubling Dimension*) *The marginal distribution μ on \mathcal{P} has a doubling dimension at most d if there exists a constant C such that for all $P \in \mathcal{P}$ and $r \geq 0$, we have $\mu(B(P, r)) \leq C\epsilon^{-d}\mu(B(P, \epsilon r))$.*

To illustrate the concept, it is instructive to look at the simple case of Euclidean space \mathbb{R}^d and uniform measure over a closed ball. In this case, it is easy to see that the doubling dimension is d . While this describes a *global* notion of doubling dimension (since it holds uniformly over all region), we will also define and use a local notion in a later section. With this setting in mind, we now analyze the consistency and convergence rates of our estimator in a rather broad framework.

4 GENERAL FRAMEWORK FOR ANALYSIS OF k -NN ESTIMATORS

In this section, we first analyze consistency and convergence rates of the k -NN based estimator in Equation (1) within a general framework. We will investigate probabilistic bounds which depend on the local intrinsic structure of the measure μ on \mathcal{P} . Finally, we develop an approach which adapts to the local intrinsic dimension by carefully choosing k . Upper bounds on the convergence rates will also be derived for this estimator.

In this general framework, we assume certain application specific bounds on the estimation of \hat{P}_i . In particular, we assume the following:

- (i) $\mathbb{E}[\rho(P_i, \hat{P}_i)] \leq \Delta$.
- (ii) $\mathbb{P}(\rho(P_i, \hat{P}_i) - \mathbb{E}[\rho(P_i, \hat{P}_i)] > \epsilon) \leq \psi_i(\epsilon)$.

These bounds Δ and ψ_i will be instantiated for the cases of distribution and function regression in later sections. The first term provides an upper bound on how close P_i is to \hat{P}_i in expected sense, while the second term measures how far the random variable $\rho(P_i, \hat{P}_i)$ is from its expected value.

4.1 CONSISTENCY OF ESTIMATOR

In this section we analyze the L_2 -consistency of the k -NN estimator in Equation (1). In order to prove the consistency of our estimator, we assume that $k \rightarrow \infty$ but $k/m \rightarrow 0$ as $m \rightarrow \infty$. This assumption is typical in k -NN like estimators [7]. Additionally, we also assume that $\Delta \rightarrow 0$ as $m \rightarrow \infty$. Using these assumptions, we prove the following consistency result.

Theorem 1. *Suppose $k \rightarrow \infty$ and $\lim_{m \rightarrow \infty} k/m = 0$. Furthermore, we assume that $\Delta \rightarrow 0$ as $m \rightarrow \infty$. Then \hat{f} is consistent, i.e., $\lim_{m \rightarrow \infty} \mathbb{E}[|\hat{f}(P) - f(P)|^2] = 0$.*

Proof. The proof is in the appendix. \square

4.2 CONVERGENCE RATES

We now turn our attention towards the convergence rates of the estimator. In particular, we prove that if the measure μ has finite doubling dimension, then we can get a nonparametric convergence rate that depends on this dimension. We already know that $\mathbb{E}[\rho(\hat{P}_i, P_i)]$ is bounded by Δ . Lemma 5 in the Supplementary material provides a bound for uniform convergence.

Let $\Omega(\epsilon_0)$ denote the event that $\rho(\hat{P}_i, P_i) \leq \Delta + \epsilon_0$ for all $i \in [m]$. From Lemma 5, we know that this event occurs with at least probability $1 - \sum_i \psi_i(\epsilon_0)$. For feasibility, we assume that ϵ_0 is large enough such that $\sum_i \psi_i(\epsilon_0) < 1$.

Theorem 2. *Let $d \geq 3$, $m' = \lfloor \frac{m}{k} \rfloor$, and Δ, ϵ_0 be such that $\nu^{-d} \leq m' \leq (4(\Delta + \epsilon_0))^{-d}$ and $\sum_i \psi_i(\epsilon_0) \leq \frac{1}{m} \nu^{-d}$. Then the following holds:*

$$\mathbb{E}[|\hat{f}(P) - f(P)|^2] \leq \frac{\sigma^2}{k} + 2C' L^2 m'^{-2/d} + 2L^2 \Delta^2.$$

for some constant C' .

Proof. The proof is in the appendix. \square

4.3 LOCAL INTRINSIC BOUNDS FOR ESTIMATOR

We establish probabilistic convergence bounds for our estimators in this section, building on the work of [10]. Our rates exploit the local intrinsic dimension of the measure μ .

In order to obtain uniform bounds over \mathcal{P} , we assume additional structure in our problem setting. We assume that the VC-dimension of class \mathcal{B} of balls on (\mathcal{P}, ρ) is $\nu_{\mathcal{B}}$. To capture the notion of local intrinsic dimension, let us define the following.

Definition 2. (*Local Doubling Dimension*) We say the measure μ has local doubling dimension d_l on $B(P, r)$ if we have $\mu(B(P, r')) \leq C\epsilon^{-d_l} \mu(B(P, \epsilon r'))$ for all $r' \leq r$ and $0 < \epsilon < 1$.

Additionally, similar to [10], we assume a noise model that has uniformly bounded tails and variance. More formally, we have for all $\delta > 0$, there exists $t > 0$ such that $\sup_{P \in \mathcal{P}} \mathbb{P}_{Y|P}(|Y - f(P)| > t) \leq \delta$. Infimum amongst all such t is denoted by $T(\delta)$. Our goal is to obtain a probabilistic upper bound on $|\hat{f}(P) - f(P)|$.

The proof uses results from [10] with additional complexity arising due to the estimation error in the variables themselves. We capture the notion of local intrinsic dimension by the doubling dimension at the neighborhood of the point P . We have the following result for the consistency of the estimator:

Theorem 3. *Suppose μ has local doubling dimension d_l on $B(P, r)$. Let $\epsilon = (3Ck/m\mu(B(P, r)))^{1/d_l}$ and $\alpha_m = (\nu_{\mathcal{B}} \log(2m) + \log(16/\delta))/m$. Also, let $\delta' = \delta + \sum_i \psi_i(\epsilon_0)$. Suppose $\mu(B(P, \epsilon r)) \geq \epsilon^{d_l} \mu(B(P, r))/C \geq 3k/m$. Then the following holds with uniformly over all $P \in \mathcal{P}$ with probability at least $1 - \delta'$,*

$$|\hat{f}(P) - f(P)|^2 \leq \frac{2\nu_{\mathcal{B}} T^2(\delta/4m) \log(4m/\delta) + 16\sigma^2}{k} + 2L^2(r + 2(\Delta + \epsilon_0))^2 \left(\frac{3Ck}{m\mu(B(P, r))} \right)^{2/d_l}.$$

Proof. Let $\tilde{f}(P) = \sum_{i=1}^m W_i f(P_i)$. Using $\tilde{f}(P)$, we get the following:

$$|\hat{f}(P) - f(P)|^2 \leq 2 \underbrace{|\hat{f}(P) - \tilde{f}(P)|^2}_{\text{Variance}} + 2 \underbrace{|\tilde{f}(P) - f(P)|^2}_{\text{Bias}}$$

This is obtained by simple application of AM-GM inequality. We first derive an upper bound for the bias and then deal with the variance. We have:

$$\begin{aligned} |\tilde{f}(P) - f(P)| &= \left| \sum_{i=1}^m W_i (f(P_i) - f(P)) \right| \\ &\leq \sum_{i=1}^m W_i |f(P_i) - f(P)| \leq L \sum_{i=1}^m W_i \rho(P, P_i). \end{aligned}$$

The first step follows from the fact that $\sum_{i=1}^m W_i = 1$. The second and third steps follow from triangle inequality and Lipschitz continuity of f respectively. Consider the index set $\mathcal{J} = \{i_1, \dots, i_k\}$ which represents the nearest neighbors of P amongst $\{\hat{P}_1, \dots, \hat{P}_m\}$ where i_j is used to denote the index of the j^{th} nearest neighbor of P amongst $\{\hat{P}_1, \dots, \hat{P}_m\}$. Similarly, let us denote by i'_j and \mathcal{J}' , the index of j^{th} nearest neighbor of P amongst $\{P_1, \dots, P_m\}$ and corresponding index set respectively. Furthermore, we use r_k to

denote $\rho(P, P_{i'_k})$. In order to obtain an upper bound on the bias we need to analyze $\max_{j \in \mathcal{J}} \rho(P, P_j)$.

Let \mathcal{E} represent the event that $\max_i D(P_i, \hat{P}_i) \leq \Delta + \epsilon_0$ for all $i \in [m]$. We know that this event occurs with probability at least $1 - \sum_i \psi_i(\epsilon_0)$. Conditioned on the event \mathcal{E} , we have the following:

$$\rho(P, \hat{P}_i) \leq \rho(P, P_i) + \rho(P_i, \hat{P}_i) \leq \rho(P, P_i) + \Delta + \epsilon_0.$$

for all $i \in [m]$. This holds due to triangle inequality and the definition of the event \mathcal{E} . Let By similar argument, we also have $\rho(P, P_i) \leq \rho(P, \hat{P}_i) + \Delta + \epsilon_0$.

The rest of the argument is conditioned on the event \mathcal{E} . Using the above relation, we get the following: $\max_{i \in \mathcal{J}'} \rho(P, \hat{P}_i) \leq r_k + \Delta + \epsilon_0$. This is due to the fact that $\rho(P, \hat{P}_i) \leq \rho(P, P_i) + \Delta + \epsilon_0$ for all $i \in [m]$ and the definition of r_k . Using the above relation, we get the following inequality:

$$\max_{i \in \mathcal{J}} \rho(P, \hat{P}_i) \leq \max_{i \in \mathcal{J}'} \rho(P, \hat{P}_i) \leq r_k + \Delta + \epsilon_0.$$

The first step holds since \mathcal{J} are the indices for the k nearest neighbors of P amongst $\{\hat{P}_1, \dots, \hat{P}_m\}$. But we also have $\rho(P, P_i) \leq \rho(P, \hat{P}_i) + (\Delta + \epsilon_0)$ for all $i \in [m]$ (since we condition on \mathcal{E}). From above argument, the following holds:

$$\begin{aligned} \max_{i \in \mathcal{J}} \rho(P, P_i) &\leq \max_{i \in \mathcal{J}} \rho(P, \hat{P}_i) + (\Delta + \epsilon_0) \\ &\leq r_k + 2(\Delta + \epsilon_0). \end{aligned}$$

In order to complete our analysis for the bias, we need to bound the distance r_k . To this end, we appeal to the bound obtained in [10]. In particular, since $\mu(B(P, \epsilon r)) \geq 3k/m$, by invoking Lemma 10 we have $\mu_m(B(P, \epsilon r)) \geq k/m$. Therefore, with probability at least $1 - \delta$, we have $r_k \leq \epsilon r$. Finally, by using union bound over the event above and \mathcal{E} , we get the required bound on the bias.

To establish a bound on the variance, we resort to the bound from [10]. We derive the bounds here for the sake of completeness. We need to bound the term $|\hat{f}(P) - \tilde{f}(P)| = |\sum_{i=1}^m W_i(Y_i - f(P_i))|$. The key step is to utilize the classical VC-theory to obtain a bound on $|Y_i - f(P_i)|$.

More formally, let us first condition on the $\mathbf{P} = \{P_1, \dots, P_m\}$. By using the concept of VC-dimension and applying union bound, we can obtain the final result. We further restrict our attention to the event where $|Y_i - f(P_i)| < T(\delta_0)$ for all $i \in [m]$. Note that this event occurs with probability at least $1 - m\delta_0 > 0.5$. This is obtained by and simple application of union bound. From Markov inequality, we have:

$$\begin{aligned} \mathbb{P}(\exists P \text{ s.t. } |\hat{f}(P) - \tilde{f}(P)| > 2\mathbb{E}(|\hat{f}(P) - \tilde{f}(P)|) + \epsilon) &\leq \\ \mathbb{P}(\exists P \text{ s.t. } |\hat{f}(P) - \tilde{f}(P)| > \mathbb{E}(|\hat{f}(P) - \tilde{f}(P)|) + \epsilon|\mathcal{E}) & \end{aligned}$$

Also note that,

$$\begin{aligned} \mathbb{P}(\exists P \text{ s.t. } |\hat{f}(P) - \tilde{f}(P)| > \mathbb{E}(|\hat{f}(P) - \tilde{f}(P)|) + \epsilon|\mathcal{E}) \\ \leq n^{\nu_B} \exp(-2k\epsilon^2/T^2(\delta_0)) \end{aligned}$$

This is due to the following facts: (i) changing any of the Y_i 's changes the function $|\hat{f}(P) - \tilde{f}(P)|$ by at most $T(\delta_0)/k$ and (ii) VC-dimension of the class of balls \mathcal{B} over \mathcal{P} is ν_B . Hence, using McDiarmid's inequality and union bound we get the above result. Let us take $\delta_0 = \delta/2m$. Now using a union bound over aforementioned events and rewriting the result using AM-GM inequality, we have with probability at least $1 - \delta$

$$\begin{aligned} |\hat{f}(P) - \tilde{f}(P)|^2 &< 8\mathbb{E}[|\hat{f}(P) - \tilde{f}(P)|^2] \\ &+ \frac{T^2(\delta/2m)}{k} (\nu_B \log(2m/\delta)) \end{aligned}$$

To complete the proof, we need to obtain an upper bound on the expected value $\mathbb{E}[|\hat{f}(P) - \tilde{f}(P)|^2]$. This is obtained in the following manner:

$$\begin{aligned} \mathbb{E}[|\hat{f}(P) - \tilde{f}(P)|^2] &= \mathbb{E}[\left| \sum_i (W_i Y_i - f(P_i)) \right|^2] \\ &= \sum_i W_i^2 \mathbb{E}[|Y_i - f(P_i)|^2] \leq \sigma^2/k \end{aligned}$$

The second equality is obtained from the fact that $Y_i - f(P_i)$ are i.i.d random variables. The last inequality is obtained from the assumption that variance of $Y|P$ is bounded by σ^2 . Combining, the bounds obtained for the bias and variance, we get the required result. \square

Note the dependence of bounds on d_l , the local doubling dimension rather than d . When $d_l \ll d$, we have obtain much better rates of convergence locally.

4.4 SELECTION OF k AND ADAPTIVE CONVERGENCE RATES

In the previous section, we obtained convergence guarantees which depend on the local intrinsic dimension for k -NN estimators. A natural question to investigate is whether these bounds provide any principled approach to choose k . Intuitively, we can see that such an approach should respect the local structure at the query point P . We derive an approach to choose k by handling the bias-variance tradeoff. Here, Δ and

Algorithm 1 Adaptive Selection of k

- 1: Let $\theta \leq \log(4m/\delta)$, Δ and ϵ_0 be given.
 - 2: Let \hat{r}_i be i^{th} nearest neighbor amongst $\{\hat{P}_1, \dots, \hat{P}_m\}$.
 - 3: $k = \arg \min_i (\theta/i + \hat{r}_i^2 + 16(\Delta + \epsilon_0)^2 \hat{r}_i)$.
-

ϵ_0 are parameters to the algorithm. Δ can be obtained through upper bound on the error or through

estimation procedures. Intuitively, the above approach can be seen as choosing k which minimizes our upper bound. We make this intuition more formal by the following result.

Theorem 4. *Suppose μ has local doubling dimension d_l on $B(P, r)$. Suppose k is chosen according to Algorithm 1 for each $P \in \mathcal{P}$ and $\hat{f}(P)$ is the k -NN estimate. Assume $((\nu_B \log(2m) + \log(16/\delta))/\theta < m^{4/(6+3d_l)})$. Let $\delta' = \delta + \sum_i \psi_i(\epsilon_0)$. Furthermore, let $r < R$ and $\mu(B(P, r)) > 6Cm^{-1/3}$. Then the following statement holds with probability at least $1 - \delta'$ simultaneously for all $P \in \mathcal{P}$.*

$$|\hat{f}(P) - f(P)|^2 \leq 2 \left(\frac{C}{\theta} + L^2 \right) \times \left((1 + 16R^2) \left(\frac{3C\theta}{m\mu(B(P, r))} \right)^{\frac{2}{2+d_l}} + 48(\Delta + \epsilon_0)^2 \right)$$

where $C = \nu_B T^2(\delta/4m) \log(4m/\delta) + 8\sigma^2$.

Proof. We have the following bound on $|\hat{f}(P) - f(P)|^2$ holds with probability at least $1 - \delta$:

$$\begin{aligned} |\hat{f}(P) - f(P)|^2 &\leq \frac{2C}{k} + 2L^2(r_k + 2(\Delta + \epsilon_0))^2 \\ &\leq \frac{2C}{k} + 2L^2(\hat{r}_k + 4(\Delta + \epsilon_0))^2 \\ &\leq \left(\frac{2C}{\theta} + 2L^2 \right) \left(\frac{\theta}{k} + (\hat{r}_k + 4(\Delta + \epsilon_0))^2 \right) \end{aligned}$$

The first and second inequalities holds from Theorem 3 and the fact that $|r_k - \hat{r}_k| \leq 2(\Delta + \epsilon_0)$ where r_k and \hat{r}_k denote the distance of k^{th} nearest neighbor of P amongst $\{P_1, \dots, P_m\}$ and $\{\hat{P}_1, \dots, \hat{P}_m\}$ respectively. Note that the procedure we use exactly minimizes $\frac{\theta}{k} + (\hat{r}_k + 4(\Delta + \epsilon_0))^2$. In order to complete the proof, we need to derive an upper bound for the estimator, we need to provide a bound on the minimum value of $\frac{\theta}{k} + (\hat{r}_k + 4(\Delta + \epsilon_0))^2$. To this end, we borrow ideas from [10] (Theorem 3), which provides a upper bound on the minimum value by explicitly constructing a k that has low objective value. We provide all the details here for sake of completeness.

Let $\tau = \theta^{d_l/(2+d_l)} \left(\frac{m\mu(B(P, r))}{3C} \right)^{2/(2+d_l)}$ Using our assumption on local doubling dimension, we have $\mu(B(P, r)) > 6C\theta m^{-d_l/(2+d_l)} \geq 6C\tau/m$. Let $\epsilon = \left(\frac{3C\tau}{m\mu(B(P, r))} \right)^{1/d_l}$. It is easy to see that from the above relationship that $\epsilon < 1$. Moreover, we have

$$\begin{aligned} \mu(B(P, \epsilon r)) &\geq \epsilon^{d_l} \mu(B(P, r))/C \geq 3\tau/m, \\ \alpha_m &= (\nu_B \log(2m) + \log(8/\delta))/m \leq \frac{\theta}{m} m^{4/6+3d_l} \\ &\leq \frac{\theta}{m} m^{4/6+3d_l} \leq \frac{\tau}{m}. \end{aligned}$$

Therefore, using Lemma 10, we have $\mu_m(B(P, \epsilon r)) \geq \frac{\tau}{m}$ with probability at least $1 - \delta$. This in turn implies that $r_k \leq \epsilon r$ for all $k \leq \tau$.

The following argument shows a bound on r_k^2 . First, observe that if $k \leq \tau$, we have

$$r_k^2 \leq (\epsilon r)^2 \leq \left(\frac{3C\tau}{m\mu(B(P, r))} \right)^{2/d_l} R^2 = \frac{R^2\theta}{\tau} \leq \frac{R^2\theta}{k}.$$

The first and last inequalities holds since $k \leq \tau$. Let k_0 be the highest integer for which the above inequality holds. It can be proved that that either k_0 or $k_0 + 1$ is larger than τ . If $k_0 > \tau$, then the above statement is obviously true. For the case of $k_0 \leq \tau$, it is easy to see that $k_0 + 1 > \tau$ since k_0 is the highest integer for which $r_k^2 \leq \frac{R^2\theta}{k}$ and this holds for all $k \leq \tau$.

Suppose $k_0 \leq \tau$. Let $k_1 = k_0 + 1$ then $\theta/(k_1) < \epsilon^2$ since $\theta/k_1 < \theta/\tau = \epsilon^2$ when $k_1 > \tau$. Moreover, $r_{k_1} \leq 2^{1/d_l} \epsilon r$ since $\mu(B(P, 2^{1/d_l} \epsilon r)) \geq 6\tau/m$ which in turn implies $\mu_m(B(P, 2^{1/d_l} \epsilon r)) \geq 2\tau/m \geq k_1/m$ (by Lemma 10 and the fact that $k_1 \leq 2\tau$).

In the other case of $k_0 > \tau$, by similar argument, we can prove that $\theta/k_0 < \theta/\tau = \epsilon^2$ and $r_{k_0}^2 \leq R^2\epsilon^2$.

Therefore, either k_0 or k_1 , satisfy the following:

$$(\theta/k + 4r_k^2) \leq (1 + 16R^2)\epsilon^2 \quad (3)$$

Since k is chosen in such a way that it minimizes $(\theta/k + (\hat{r}_k + 4(\Delta + \epsilon_0))^2)$, we have the following bound:

$$\begin{aligned} |\hat{f}(P) - f(P)|^2 &\leq \left(\frac{2C}{\theta} + 2L^2 \right) \left(\frac{\theta}{k} + (\hat{r}_k + 4(\Delta + \epsilon_0))^2 \right) \\ &\leq \min_{k_0, k_1} \left(\frac{2C}{\theta} + 2L^2 \right) \left(\frac{\theta}{k} + (\hat{r}_k + 4(\Delta + \epsilon_0))^2 \right) \\ &\leq \min_{k_0, k_1} \left(\frac{2C}{\theta} + 2L^2 \right) \left(\frac{\theta}{k} + (2\hat{r}_k^2 + 32(\Delta + \epsilon_0)^2) \right) \\ &\leq \left(\frac{2C}{\theta} + 2L^2 \right) ((1 + 16R^2)\epsilon^2 + 48(\Delta + \epsilon_0)^2) \end{aligned}$$

The first and second inequalities follow from the fact that $|\hat{r}_k - r_k| \leq 2(\Delta + \epsilon_0)$ and the criteria of choosing k . The final inequality follows from Equation (3). This gives us the required result. \square

The above result shows adaptive convergence rates for the k -NN estimators. We now proceed towards applications of the general framework we just discussed.

5 APPLICATIONS

We discuss specific applications of the general framework introduced in the previous section. More specifically, we look at *distribution* regression and *function*

regression settings. We will see that by using appropriate instantiation of the bounds Δ and ψ_i in the general framework, the results for both these case follow in a straightforward manner.

5.1 DISTRIBUTION REGRESSION

We describe distribution regression problem in this section. We consider a regression problem where the input variables are from the space of continuous 1-Lipschitz probability distributions (i.e., $|P(x) - P(y)| \leq \|x - y\|$) on a compact subset $\mathcal{K} \subset \mathbb{R}^p$ (denoted by \mathcal{D}). In this case, $\mathcal{P} = \mathcal{D}$ and we assume ρ to be the L_1 distance between distributions, i.e., $\rho(P, Q) = \|P - Q\|_1 = \int |P(x) - Q(x)| dx$. Note that $f : \mathcal{D} \rightarrow \mathbb{R}$. We assume that class of balls \mathcal{B} on \mathcal{D} have finite VC-dimension $\nu_{\mathcal{B}}$. Here, we have $Y_i = f(P_i) + \gamma_i$ for all $i \in [m]$.

The measurement error comes into play due to the fact that we do not have access to the probability distributions P_i directly; rather we observe samples $X_{i1}, \dots, X_{in_i} \sim P_i$. From these samples, we estimate the probability distributions through one of the several density estimation procedures like kernel density estimation, data clustering. Let $\hat{P}_1, \dots, \hat{P}_m$ be estimated probability distributions corresponding to P_1, \dots, P_m respectively. To summarize, we think of observations as $(\mathbf{P}, \mathbf{Y}) = \{(\hat{P}_i, Y_i)\}_{i=1}^m$ and our goal is to infer the function f . For ease of exposition, we assume that the number of samples observed for all the distributions, i.e., $n = n_i$ for all $i \in [m]$. To apply our framework, we need to instantiate the bounds Δ and ψ_i .

Bound Δ : We have the following bound on the expected error of estimation of the distributions.

Lemma 1. *Under above conditions, we have*

$$\mathbb{E}[\rho(\hat{P}_i, P_i)] \leq \tilde{C} n^{-1/(2+p)}$$

where \tilde{C} is a constant.

Using this result we can take $\Delta = \tilde{C} n^{-1/(2+p)}$. Refer [19] for details of the proof.

Bound ψ_i : We obtain the following bound ψ_i by using McDiarmid's inequality.

Lemma 2. *Under above conditions, we have*

$$\mathbb{P}(\rho(\hat{P}_i, P_i) > \mathbb{E}[\rho(\hat{P}_i, P_i)] + \epsilon) \leq e^{-n\epsilon^2/2}$$

Therefore, $\psi_i(\epsilon) = \exp(-n\epsilon^2/2)$ (see [2] for details).

By using the above bounds, we present the main results for distribution regression. Our first result is the consistency of the estimator \hat{f} . From Theorem 1, we have the following result. We set $\Delta = \tilde{C} n^{-1/(2+p)}$ and $\epsilon_0 = n^{-1/(2+p)}$.

Theorem 5. *(Consistency of Estimator) Suppose $k \rightarrow \infty$ and $\lim_{m \rightarrow \infty} k/m = 0$. Furthermore, we assume that $n = \Omega(\log^{(2+p)/p}(m))$. Then \hat{f} is consistent, i.e., $\lim_{m \rightarrow \infty} \mathbb{E}[|\hat{f}(P) - f(P)|^2] = 0$.*

The next result provides convergence rates for distribution regression by directly appealing to Theorem 2.

Theorem 6. *(Convergence Rate) Let $d \geq 3$ and $m' = \lfloor \frac{m}{k} \rfloor$. Assume $n \geq (2 \log(m) + 2d \log(\nu))^{(2+p)/p}$ and $d \log(1/\nu) \leq \log(m') \leq d \log(n)/(2+p) - d(\log(4 + 4\tilde{C}))$. Then the following holds:*

$$\mathbb{E}[|\hat{f}(P) - f(P)|^2] \leq \frac{\sigma^2}{k} + \frac{2C'L^2}{m'^{2/d}} + 2\frac{\tilde{C}^2 L^2}{n^{2/(2+p)}}.$$

for some constant C' .

The following result shows that the convergence rates in fact depend on the local intrinsic dimension of the probability measure μ . This is obtain from Theorem 3 of the general framework.

Theorem 7. *(Adaptive Convergence Rates) Suppose μ has local doubling dimension d_l on $B(P, r)$. Let $\epsilon = (3Ck/m\mu(B(P, r)))^{1/d_l}$ and $\alpha_m = (\nu_{\mathcal{B}} \log(2m) + \log(16/\delta))/m$. Let $\delta' = \delta + m \exp(-n^{p/(p+2)}/2)$. Suppose $\mu(B(P, \epsilon r)) \geq \epsilon^{d_l} \mu(B(P, r))/C \geq 3k/m$. Then the following holds with uniformly over all $P \in \mathcal{P}$ with probability at least $1 - \delta'$,*

$$|\hat{f}(P) - f(P)|^2 \leq \frac{2\nu_{\mathcal{B}} T^2 (\delta/4m) \log(4m/\delta) + 16\sigma^2}{k} + 2L^2 \left(r + 2\frac{(\tilde{C} + 1)}{n^{1/(2+p)}} \right)^2 \left(\frac{3Ck}{m\mu(B(P, r))} \right)^{2/d_l}.$$

The final result for distribution regression shows that by using Algorithm 1, we obtain reasonable adaptive convergence rates.

Theorem 8. *(k-Selection Convergence Rates) Suppose μ has local doubling dimension d_l on $B(P, r)$. Suppose k is chosen according to Algorithm 1 for each $P \in \mathcal{P}$ and $\hat{f}(P)$ is the k -NN estimate. Assume $((\nu_{\mathcal{B}} \log(2m) + \log(16/\delta))/\theta) < m^{4/(6+3d_l)}$. Let $\delta' = \delta + m \exp(-n^{p/(p+2)}/2)$. Furthermore, let $r < R$ and $\mu(B(P, r)) > 6Cm^{-1/3}$. Then the following statement holds with probability at least $1 - \delta'$ simultaneously for all $P \in \mathcal{P}$.*

$$|\hat{f}(P) - f(P)|^2 \leq 2 \left(\frac{C_0}{\theta} + L^2 \right) \times \left((1 + 16R^2) \left(\frac{3C\theta}{m\mu(B(P, r))} \right)^{\frac{2}{2+d_l}} + \frac{48(\tilde{C} + 1)^2}{n^{-2/(2+p)}} \right)$$

where $C_0 = \nu_{\mathcal{B}} T^2 (\delta/4m) \log(4m/\delta) + 8\sigma^2$.

5.2 FUNCTION REGRESSION

In this section, we describe another interesting application—function regression. Here, the input variables belong to the class of 1-Lipschitz (w.r.t Euclidean distance) functions on $[0, 1]$ (denoted by \mathcal{F}). For this case, we use $\rho(P, Q) = \|P - Q\|_2 = \sqrt{\int_0^1 (P(x) - Q(x))^2 dx}$ (norm in L^2 space of functions). We again focus only on the case when class of balls \mathcal{B} on $\mathcal{P} = \mathcal{F}$ have finite VC-dimension $\nu_{\mathcal{B}}$. The model is $Y_i = f(P_i) + \gamma_i$ for all $i \in [m]$ where P_i are functions. Similar to the distribution regression, we usually do not have access to the function themselves but only the ability to obtain noisy estimates of function value at certain points. For simplicity, we assume a deterministic design where we query at the points $\{X_{ij}\}_{j=1}^n$ where $X_{ij} = j/n$ for all $i \in [m]$ (see [21] for more details about deterministic design). Therefore, we have $Z_{ij} = P_i(X_{ij}) + \zeta_{ij}$ where ζ_{ij} is the noise variable with $\mathbb{E}(\zeta_{ij}) = 0$ and variance $\bar{\sigma}^2$.

We can intuitively think of function regression as 2-stage regression problem. We first estimate the functions themselves and then perform another regression on these functions to obtain the functional of interest f . From $\{(X_{ij}, Z_{ij})\}_{j=1}^n$, we can obtain estimated functions $\hat{P}_1, \dots, \hat{P}_m$ corresponding to P_1, \dots, P_m respectively. This model now fits our framework perfectly once we have appropriate bounds Δ and ψ_i . To obtain these bounds, we directly appeal to the following well-known bounds for regression.

Bound Δ : We have the following bound for Δ in the case of function regression.

Lemma 3. *Under the conditions mentioned above, we have*

$$\mathbb{E}[\rho(\hat{P}_i, P_i)] \leq \bar{C}n^{-1/3}$$

where \bar{C} is a constant.

Proof. From Jensen's inequality, we have

$$\mathbb{E}[\rho(\hat{P}_i, P_i)]^2 \leq \mathbb{E}[\rho^2(\hat{P}_i, P_i)] \leq \bar{C}^2 n^{-2/3}.$$

The last inequality follows from Corollary 1.2 of [21]. \square

Bound ψ_i : To obtain bound ψ_i , we resort to McDiarmid's inequality and obtain the following bound.

Lemma 4. *Under the conditions mentioned above, we have*

$$\mathbb{P}(\rho(\hat{P}_i, P_i) > \mathbb{E}[\rho(\hat{P}_i, P_i)] + \epsilon) \leq e^{-n\epsilon^2/2}$$

Proof. The result follows from simple application of McDiarmid's inequality. \square

We now state the main results for function regression by appealing to the general framework. We set $\Delta = \bar{C}n^{-1/3}$ and $\epsilon_0 = n^{-1/3}$. The following result shows consistency of estimator \hat{f} in case of function regression. This is obtained from Theorem 1.

Theorem 9. *(Consistency of Estimator) Suppose $k \rightarrow \infty$ and $\lim_{m \rightarrow \infty} k/m = 0$. Furthermore, we assume that $n = \Omega(\log^3(m))$. Then \hat{f} is consistent, i.e., $\lim_{m \rightarrow \infty} \mathbb{E}[|\hat{f}(P) - f(P)|^2] = 0$.*

The next result provides convergence rates for function regression by directly using Theorem 2.

Theorem 10. *(Convergence Rate) Let $d \geq 3$ and $m' = \lfloor \frac{m}{k} \rfloor$. Assume $n \geq (2 \log(m) + 2d \log(\nu))^3$ and $d \log(1/\nu) \leq \log(m') \leq d \log(n)/3 - d(\log(4 + 4\bar{C}))$. Then the following holds:*

$$\mathbb{E}[|\hat{f}(P) - f(P)|^2] \leq \frac{\sigma^2}{k} + \frac{2C'L^2}{m'^{2/d}} + \frac{2\bar{C}^2 L^2}{n^{2/3}}.$$

for some constant C' .

The following result shows that the convergence rates in fact depend on the local intrinsic dimension of the probability measure μ .

Theorem 11. *(Adaptive Convergence Rates) Suppose μ has local doubling dimension d_l on $B(P, r)$. Let $\epsilon = (3Ck/m\mu(B(P, r)))^{1/d_l}$ and $\alpha_m = (\nu_{\mathcal{B}} \log(2m) + \log(16/\delta))/m$. Let $\delta' = \delta + m \exp(-n^{1/3}/2)$. Suppose $\mu(B(P, \epsilon r)) \geq \epsilon^{d_l} \mu(B(P, r))/C \geq 3k/m$. Then the following holds uniformly over all $P \in \mathcal{P}$ with probability at least $1 - \delta'$,*

$$|\hat{f}(P) - f(P)|^2 \leq \frac{2\nu_{\mathcal{B}} T^2 (\delta/4m) \log(4m/\delta) + 16\sigma^2}{k} + 2L^2 \left(r + \frac{2(\bar{C} + 1)}{n^{1/3}} \right)^2 \left(\frac{3Ck}{m\mu(B(P, r))} \right)^{2/d_l}.$$

The final result for distribution regression shows that by using Algorithm 1, we obtain reasonable adaptive convergence rates.

Theorem 12. *(k-Selection Convergence Rates) Suppose μ has local doubling dimension d_l on $B(P, r)$. Suppose k is chosen according to Algorithm 1 for each $P \in \mathcal{P}$ and $\hat{f}(P)$ is the k -NN estimate. Assume $((\nu_{\mathcal{B}} \log(2m) + \log(16/\delta))/\theta < m^{4/(6+3d_l)})$. Let $\delta' = \delta + m \exp(-n^{1/3}/2)$. Furthermore, let $r < R$ and $\mu(B(P, r)) > 6Cm^{-1/3}$. Then the following holds with probability at least $1 - \delta'$ simultaneously for all $P \in \mathcal{P}$.*

$$|\hat{f}(P) - f(P)|^2 \leq 2 \left(\frac{C_0}{\theta} + L^2 \right) \times \left((1 + 16R^2) \left(\frac{3C\theta}{m\mu(B(P, r))} \right)^{\frac{2}{2+d_l}} + \frac{48(\bar{C} + 1)^2}{n^{-2/3}} \right)$$

where $C_0 = \nu_{\mathcal{B}} T^2 (\delta/4m) \log(4m/\delta) + 8\sigma^2$.

Before ending our discussion of technical results, it is worthy to note two points: (i) Our rates are faster than the logarithmic rates that are sometimes obtained in measurement error nonparametric regression models as in [3]. As mentioned earlier, this is due to the fact that the measurement error corresponds to $\rho(\hat{P}_i, P_i)$ which is not Gaussian for finite n_i and which decreases when n_i increases. (ii) Typically, it is difficult to estimate the distance $\rho(P, Q)$ exactly. This presents additional level of complexity but it can be handled gracefully within our framework by viewing it as another measurement error.

6 EXPERIMENTS

Although the main contribution of our paper is to provide theoretical insights in k -NN based estimation for functional data, we also provide numerical evidence showing the empirical benefits of these estimators. We consider two distribution regression tasks: Beta distribution skewness and Gaussian distribution entropy estimation. In our experiments, we set all the n, n_1, \dots, n_m set sizes to the same values, which will be specified below. In the first experiment, we generated 325 sample sets from $\text{Beta}(a, 3)$ distributions where a was varied between $[3, 20]$ randomly. We constructed $m = 250$ sample sets for training, 25 for validation, and 50 for testing. Each sample set contained $n = 500$ i.i.d samples from $\text{Beta}(a, 3)$. Our task in this experiment was to learn the skewness of $\text{Beta}(a, b)$ distributions, $f = \frac{2(b-a)\sqrt{a+b+1}}{(a+b+2)\sqrt{ab}}$. We considered the noiseless case, i.e., γ was set to zero. Our estimator is oblivious of the fact that the sample sets are coming from Beta distributions, and it does not know the skewness function values in the test sets either; its values are available only in the training and validation sets.

For obtaining the empirical probability distribution, we use kernel density estimation with Gaussian kernel. The optimal bandwidth of the kernel is obtained by cross validation. To estimate the L_2 distances between \hat{p}_i and p , we calculated their estimated values in 4096 points on a uniformly distributed grid between the min and max values in the sample sets, and then estimated the integral $\int (p(x) - \hat{p}_i(x))^2 dx$ with the rectangle method for numerical integration. To find the appropriate k , we selected the value from $\{1, \dots, 10\}$ that lead to minimum MSE on validation set. Figure 1(a) displays the predicted values for the 50 test sample sets, and we also show the true values of the skewness functions. As we can see the true and the estimated values are very close to each other.

In the next experiment, our task was to learn the entropy of Gaussian distributions. We chose a 2×2 covariance matrix $\Sigma = AA^T$, where $A \in \mathbb{R}^{2 \times 2}$, and A_{ij}

was randomly selected from the uniform distribution $U[0, 1]$. Just as in the previous experiments we constructed 325 sample sets from $\{\mathcal{N}(0, R(\alpha_i)\Sigma^{1/2})\}_{i=1}^{325}$. Where $R(\alpha_i)$ is a 2d rotation matrix with rotation angle $\alpha_i = i\pi/325$. From each $\mathcal{N}(0, R(\alpha_i)\Sigma^{1/2})$ distribution we sampled 500 2-dimensional i.i.d. points. Similarly to the previous experiment, 250 points was used for training, 25 for selecting appropriate bandwidth parameters, and 50 for training. Our goal was to learn the entropy of the first marginal distribution: $f = \frac{1}{2} \ln(2\pi e \sigma^2)$, where $\sigma^2 = M_{1,1}$ and $M = R(\alpha_i)\Sigma R^T(\alpha_i) \in \mathbb{R}^{2 \times 2}$. μ was zero in this experiment as well. Figure 1(b) displays the learned entropies of the 50 test sample sets. The true and the estimated values are close to each other in this experiment as well.

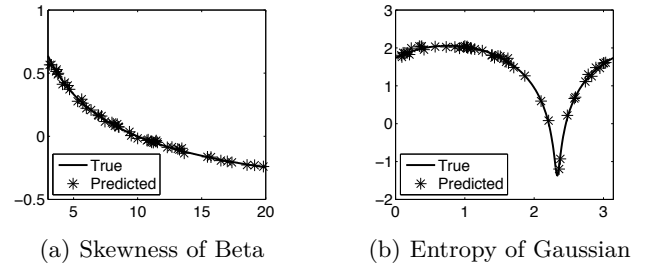


Figure 1: (a) Learned skewness of $\text{Beta}(a, 3)$ distribution. Axis x : parameter a in $[3, 20]$. Axis y : skewness of $\text{Beta}(a, 3)$. (b) Learned entropy of a 1d marginal distribution of a rotated 2d Gaussian distribution. Axes x : rotation angle in $[0, \pi]$. Axis y : entropy. The MSE in two cases are 7.1×10^{-3} and 8.6×10^{-2} respectively.

7 CONCLUSION

We presented a general framework for k -NN estimators for functional data with measurement error. We proved consistency of the estimator and derived upper bounds on the risk. We also analyzed probabilistic bounds capturing the local intrinsic dimension. Furthermore, we presented an algorithm for adaptively choosing k . Two interesting applications of our framework—distribution and function regression—were presented.

In future work, we would like to study lower bounds for the problem and compare our results with the minimax bounds. From practical point of view, it would also be interesting to use these estimators in conjunction with cover trees [1] to obtain fast k -NN estimators. Analyzing the empirical performance on large datasets is another interesting direction.

References

- [1] Alina Beygelzimer, Sham Kakade, and John Langford. Cover trees for nearest neighbor. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, 2006.
- [2] L. Devroye and G. Lugosi. *Combinatorial methods in density estimation*. Springer, 2001.
- [3] J. Fan and Y.K. Truong. Nonparametric regression with errors in variables. *The Annals of Statistics*, pages 1900–1925, 1993.
- [4] F. Ferraty and P. Vieu. *Nonparametric Functional Data Analysis: Theory and Practice*. Springer Verlag, 2006.
- [5] Alexander Gray and Andrew Moore. ‘n-body’ problems in statistical learning. In *Advances in Neural Information Processing Systems 13*, pages 521–527. MIT Press, 2000.
- [6] S. Grunewalder, G. Lever, L. Baldassarre, S. Paterson, A. Gretton, and M. Pontil. Conditional mean embeddings as regressors. In *ICML*, 2012.
- [7] L. Györfi, M. Kohler, A. Krzyzak, and H. Walk. *A Distribution-Free Theory of Nonparametric Regression*. Springer, New-york, 2002.
- [8] T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *NIPS*, pages 487–493. MIT Press, 1998.
- [9] T. Jebara, R. Kondor, A. Howard, K. Bennett, and N. Cesa-bianchi. Probability product kernels. *JMLR*, 5:819–844, 2004.
- [10] S. Kpotufe. k-nn regression adapts to local intrinsic dimension. *arXiv.org*, 2011.
- [11] Nicola Mingotti, Rosa E. Lillo, and Juan Romo. Lasso variable selection in functional regression. Statistics and econometrics working papers, May 2013.
- [12] P. Moreno, P. Ho, and N. Vasconcelos. A Kullback-Leibler divergence based kernel for SVM classification in multimedia applications. In *NIPS*, 2004.
- [13] Krikamol Muandet, Kenji Fukumizu, Francesco Dinuzzo, and Bernhard Schölkopf. Learning from distributions via support measure machines. In *NIPS*, pages 10–18, 2012.
- [14] Junier B. Oliva, Barnabás Póczos, and Jeff G. Schneider. Distribution to distribution regression. In *ICML (3)*, pages 1049–1057, 2013.
- [15] Junier B. Oliva, Barnabás Póczos, Timothy Verstynen, Aarti Singh, Jeff G. Schneider, Fang-Cheng Yeh, and Wen-Yih Isaac Tseng. Fusso: Functional shrinkage and selection operator. *CoRR*, abs/1311.2234, 2013.
- [16] Barnabás Póczos, Aarti Singh, Alessandro Rinaldo, and Larry A. Wasserman. Distribution-free distribution regression. In *AISTATS*, pages 507–515, 2013.
- [17] J.O. Ramsay and B.W Silverman. *Functional data analysis*. Springer, New York, 2nd edition, 2005.
- [18] Sashank J. Reddi and Barnabás Póczos. Scale invariant conditional dependence measures. In *Proceedings of the 28th International Conference on Machine Learning*, pages 1355–1363, 2013.
- [19] P. Rigollet and R. Vert. Optimal rates for plug-in estimators of density level sets. *Bernoulli*, 15(4):1154–1178, 2009.
- [20] A. J. Smola, A. Gretton, L. Song, and B. Schölkopf. A Hilbert space embedding for distributions. volume 4754, pages 13–31. Springer, 2007.
- [21] A.B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer, 2010.
- [22] Jeffrey K. Uhlmann. Satisfying general proximity/similarity queries with metric trees. *Information Processing Letters*, 40(4):175–179, 1991.

Bayesian Inference in Treewidth-Bounded Graphical Models Without Indegree Constraints

Daniel J. Rosenkrantz
University at Albany – SUNY
Albany, NY 12222
drosenkrantz@gmail.com

Madhav V. Marathe
Virginia Tech
Blacksburg, VA 24061
mmarathe@vbi.vt.edu

S. S. Ravi
University at Albany – SUNY
Albany, NY 12222
ravi@cs.albany.edu

Anil K. Vullikanti
Virginia Tech
Blacksburg, VA 24061
akumar@vbi.vt.edu

Abstract

We present new polynomial time algorithms for inference problems in Bayesian networks (BNs) when restricted to instances that satisfy the following two conditions: they have bounded treewidth and the conditional probability table (CPT) at each node is specified concisely using an *r-symmetric function* for some constant *r*. Our polynomial time algorithms work directly on the *unmoralized* graph. Our results significantly extend known results regarding inference problems on treewidth bounded BNs to a larger class of problem instances. We also show that relaxing either of the conditions used by our algorithms leads to computational intractability.

1 INTRODUCTION

Bayesian Networks (BNs) represent dependencies among a collection of probabilistic domain variables (Darwiche, 2009; Koller and Friedman, 2009; Pearl, 1988). Structurally, a BN $G(V, E)$ is a *directed acyclic graph* (dag) in which each node $v \in V$ represents a stochastic variable x_v ; a directed edge (u, v) in E indicates that variable x_v depends on x_u . Each node v is also associated with a table which gives the probability distribution of x_v conditioned on the variables on which x_v depends. Thus, a BN provides a simple graphical representation of the dependencies among domain variables.

BNs can be used to formulate and solve many problems in the context of stochastic decision support systems. For example, in the **inference problem**, the input is an observation (i.e., the observed values of a nonempty subset of variables) and the goal is to compute the conditional probability distribution for one specified variable. Such problems are useful in many application domains including medical diagnosis, weather forecasting, design of diagnosis-and-repair modules in computer systems, etc. (Darwiche, 2009; Koller and Friedman, 2009; Pearl, 1988).

Formal definitions of inference problems for BNs are provided in Section 2. In general, obtaining exact or approximate solutions to these problems is known to be computationally intractable (Abdelbar et al. (2000); Cooper (1990); Dagum and Luby (1993); Darwiche (2009); de Campos (2011)). Given the practical importance of these problems, researchers have tried to identify restricted versions of the problems which are useful in practice and which can be solved in polynomial time. An important development in this direction is the result of Lauritzen and Spiegelhalter (1988) who showed that for BNs of bounded treewidth, the inference problems can be solved in polynomial time using dynamic programming. Their approach uses the *moralized* form of the network, where for any node v , the parents of v are connected together as a clique. As a consequence, a moralized BN has bounded treewidth only when the maximum indegree in the unmoralized BN is also bounded.

Our Contributions: Our main result is a new dynamic programming approach that extends the class of BNs for which various inference problems can be solved in polynomial time. In particular, our approach does *not* use moralization. Instead, it works with the given BN and its tree decomposition. Thus, our algorithms are applicable to treewidth-bounded BNs, even when the indegrees of nodes are not bounded. Allowing nodes of unbounded indegree introduces a difficulty, namely that a fully specified CPT at a node may be exponentially large. To overcome this difficulty, we require the conditional probability tables at each node to be specified concisely using certain restricted classes of functions, called *r-symmetric functions* for some fixed integer *r*. As will be explained in Section 2, any CPT for a BN with maximum indegree *d* can be specified as a *d-symmetric function*. In other words, CPTs for BNs with bounded indegrees are a restricted form of *r-symmetric functions*. Thus, our approach identifies a larger class of BNs for which inference problems can be solved efficiently. Our results also extend the earlier results in (Bacchus et al., 2003; Courcelle et al., 2001; Fischer et al., 2008; Samer and Szeider, 2007) as discussed below. The results in (Courcelle et al., 2001; Fischer et al., 2008; Samer and Szeider, 2007) when combined with those of (Bacchus

et al., 2003) can be used to obtain polynomial time results for treewidth bounded BNs, with no a priori bound on indegree but whose CPTs are specified using certain threshold functions. As will be explained in Section 2.3, the class of r -symmetric functions is a strict superset of the class of threshold functions.

We also present hardness results that provide an indication of the tightness of our efficient solvability results. In particular, we show that if the conditional probability tables are not necessarily r -symmetric, then the inference problems remain computationally intractable (#P-hard) even when the BN is a directed tree (whose treewidth is 1). We also show that if the treewidth of the BN is not bounded, the inference problems remain computationally intractable even when each conditional probability table is expressed as a symmetric function. In other words, relaxing either of the assumptions (treewidth boundedness or r -symmetric functions) makes the problems computationally intractable. We note that the necessity of bounded treewidth for efficient inference in BNs was proven in Kwisthout et al. (2010) under a different assumption regarding complexity classes, namely, the Exponential Time Hypothesis (Impagliazzo and Paturi, 2001).

The remainder of this paper is organized as follows. Section 2 defines the necessary graph theoretic terms and presents formulations of several computations problems for BNs. It also discusses some related work on these problems. For space reasons, Section 3 presents our algorithm assuming that each CPT is a 1-symmetric function. Extensions of the result to other inference problems and r -symmetric functions (for any $r \geq 2$) are discussed in a longer version of this paper (Rosenkrantz et al., 2014). Section 4 mentions our hardness results whose proofs also appear in Rosenkrantz et al. (2014). Section 5 summarizes the paper and provides directions for future work.

2 DEFINITIONS AND PREVIOUS WORK

2.1 BAYESIAN NETWORKS

As mentioned earlier, a Bayesian network (BN) consists of a directed acyclic graph $G(V, E)$, where nodes represent stochastic domain variables and directed edges represent dependencies between variables. For simplicity, we will assume that each node represents a Boolean variable; the results in this paper can be extended to variables that assume values from a finite domain. Also, we do not distinguish between a node of the graph and the corresponding Boolean variable. When there is a directed edge $(u, v) \in E$, we say that u is a **parent** of v . The **indegree** of a node v is the number of parents of v .

At each node v , there is a **conditional probability table** (CPT) T_v which specifies the probability values for the variable v , conditioned on the parents of v . For a node v with indegree t , there are 2^t different combinations of

Boolean values for the parents of v . For each such combination, the table specifies the probability of v being 1 (or 0) conditioned on the parents assuming the given combination of values. Thus, the number of entries in T_v is 2^t . For a node v with indegree 0 (i.e., a node which does not have any parent), the table T_v specifies simply the probability of v assuming the value 1 (or 0).

Our formulation of the computational problems for BNs follows the presentation in (Bodlaender, 2004). For a node v , let $\mathcal{P}(v)$ denote the set of parents of v . Given a BN $G(V, E)$, a **configuration** c_V is an assignment of Boolean values to each variable in V . Given a subset $O \subseteq V$, a **partial configuration** c_O on O specifies a value for each variable in O . Given a configuration c_V (or a partial configuration c_O), we use $c_V(v)$ ($c_O(v)$) to denote the value of variable v in that configuration (partial configuration). With a slight abuse of notation, we also extend this notation to subsets of variables. Thus, given a configuration c_V (or a partial configuration c_O), and a subset $W \subseteq V$ ($W \subseteq O$), $c_V(W)$ ($c_O(W)$) denotes the combination of values assigned to the variables in W . Given a configuration c_V , its probability $\Pr\{c_V\}$ is given by

$$\Pr\{c_V\} = \prod_{v \in V} \Pr\{c_V(v) \mid c_V(\mathcal{P}(v))\}.$$

A configuration c_V is an **extension** of a partial configuration c_O if for each variable v that is assigned a value in c_O , $c_V(v) = c_O(v)$. Thus, an extension of a partial configuration c_O is obtained by specifying values for the variables that are not assigned a value in c_O .

We now provide formal definitions of two commonly considered problems in the context of BNs. In all these problems, we are given a partial configuration c_O (also called an **observation**) on a set of nodes $O \subseteq V$.

Definition 2.1 Let $G(V, E)$ denote the given BN.

1. **Inference Problem** (denoted by INF): Given an observation c_O and a variable v , find $\Pr\{v = 1 \mid c_O\}$, that is, the probability that v assumes the value 1 conditioned on the observation c_O .
2. **Most Probable Explanation Problem** (denoted by MPE): Given an observation c_O , find an extension of c_O which has the maximum probability among all the extensions of c_O .

For ease of exposition, we will also consider the following problem, which we call the **Probability Computation Problem** (denoted by PROB): Given an observation c_O , find the probability of c_O , that is, the sum of the probabilities of all configurations that are extensions of c_O . (Thus, when an observation does not specify a value for any variable, the answer to the PROB problem is 1.) The reasons for considering the PROB problem are twofold. First, the solution to INF problem for a node v and observation c_O can

be obtained using two calls to the PROB problem: compute $\Pr\{c_O\}$ and $\Pr\{v = 1 \wedge c_O\}$ using the algorithm for PROB and use the fact that

$$\Pr\{v = 1 \mid c_O\} = \frac{\Pr\{v = 1 \wedge c_O\}}{\Pr\{c_O\}}.$$

Second, an algorithm for the MPE problem can be devised along lines that are similar to that for the PROB problem. The main modification is that while the algorithm for the PROB problem computes sums of probability values at various steps, the algorithm for the MPE problem computes the maximum of the probability values.

2.2 TREE DECOMPOSITIONS

We now recall the standard definition of *tree decomposition* and *treewidth* from (Bodlaender, 1993), which will be used throughout this paper.

Definition 2.2 Given a BN $G(V, E)$, a **tree decomposition** of G is a pair $(\{X_i \mid i \in I\}, T = (I, F))$, where $\{X_i \mid i \in I\}$ is a family of subsets of V and $T = (I, F)$ is an undirected tree with the following properties:

1. $\bigcup_{i \in I} X_i = V$.
2. For every directed edge $e = (v, w) \in E$, there is a subset X_i , $i \in I$, with $v \in X_i$ and $w \in X_i$.
3. For all $i, j, k \in I$, if j lies on the path from i to k in T , then $X_i \cap X_k \subseteq X_j$.

The **treewidth** of a tree decomposition $(\{X_i \mid i \in I\}, T)$ is $\max_{i \in I} \{|X_i| - 1\}$. The treewidth of a graph is the minimum over the treewidths of all its tree decompositions.

A class of graphs is **treewidth bounded** if there is a constant k such that the treewidth of every graph in the class is at most k .

A number of problems that are NP-hard on general graphs can be solved efficiently when restricted to the class of treewidth-bounded graphs. A considerable amount of work has been done in this area (see for example (Bodlaender, 1997, 1993; Courcelle and Mosbah, 1993; Gottlob and Szeider, 2008; Robertson and Seymour, 1986) and the references therein).

As mentioned earlier, our approach works on the given (unmoralized) BN. To illustrate the effect of moralization on a BN, consider the class of directed star graphs defined as follows. For each $n \geq 2$, a directed star graph has n nodes and $n - 1$ directed edges; there is one center node and each of the other $n - 1$ nodes has just one outgoing edge to the center node. Thus, the center node has $n - 1$ parents and the moralized graph has a clique of size $n - 1$. Consequently, the moralized graph is not treewidth-bounded. On the other hand, it can be seen that according to Definition 2.2, this class of graphs has a treewidth of 1.

2.3 SPECIFYING CPTs CONCISELY

For a node v with q parents, the CPT T_v has 2^q entries. For BNs in which the maximum indegree is bounded, the CPTs can be given explicitly, since the size of each table is just a constant. However, when we consider BNs in which node indegrees may not be bounded, the size of a fully specified table may be exponential in the size of the BN. Thus, we need a method of specifying the CPTs concisely. We do this by identifying restricted classes of functions to specify the tables.

Consider a node v with q parents w_1, w_2, \dots, w_q . A CPT T_v for v specifies a probability value (i.e., the value $\Pr\{v = 1\}$) for each combination of values of the parents of v . Thus, T_v represents a function from $\{0, 1\}^q$ to the set of real values in $[0, 1]$. By restricting the class of functions, we can specify T_v concisely. We will now present some examples of such restrictions.

Definition 2.3 Let q be an integer ≥ 1 . A function f from $\{0, 1\}^q$ to the set of real values in $[0, 1]$ is said to be **symmetric** if the value of f depends only on the number of inputs which are 1.

Thus, a symmetric function f of q variables can be concisely described by specifying $q + 1$ probability values p_0, p_1, \dots, p_q , where p_i is the probability value when i of the inputs are 1, $0 \leq i \leq q$.

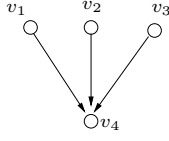
Example: Consider the BN shown in Figure 1. In that figure, nodes v_1, v_2 and v_3 don't have any parents. So, the probability values assigned to them can be thought of as symmetric functions where the only possible value for the number of parents is zero. Node v_4 has three parents. Hence, the CPT for v_4 shows the value of $\Pr\{v_4 = 1\}$ when 0, 1, 2 or 3 parents of v_4 are assigned the value 1. \square

For any integer $t \geq 0$, a **t -threshold** Boolean function on q inputs takes on the value 1 iff at least t of the inputs are 1. It is easy to see that each t -threshold function is a symmetric function. Thus, the class of symmetric functions contains all threshold functions.

One can define a further generalization of the class of symmetric functions as follows (Barrett et al., 2007b).

Definition 2.4 Let $r \geq 1$ be a fixed integer. Let q be an integer ≥ 1 . A function f from $\{0, 1\}^q$ to the set of real values in $[0, 1]$ is said to be **r -symmetric** if the set of inputs can be partitioned into r classes such that the value of f depends only on the number of 1-valued inputs in each class.

Note that any symmetric function is 1-symmetric. It can also be seen that for any $r \geq 1$, any r -symmetric function f of q variables can be concisely described by specifying $O(q^r)$ probability values. Since r is fixed, the size of the specification of any r -symmetric function is a polynomial in the size of the BN.



$$\Pr\{v_1 = 1\} = \Pr\{v_2 = 1\} = \Pr\{v_3 = 1\} = 1/2$$

CPT for node v_4 :

$ \mathcal{P}(v_4) $	$\Pr\{v_4 = 1 \mid \mathcal{P}(v_4)\}$
0	1/2
1	1/3
2	1/4
3	1/5

Figure 1: An Example of a BN where each CPT is a symmetric function. (Recall that $\mathcal{P}(v_4)$ denotes the set of parents of node v_4 .)

When a node has a bounded indegree, say d , the corresponding CPT can be thought of as a d -symmetric function, where each of the d classes contains exactly one input. Thus, the class of BNs in which each node has a bounded indegree is a special case of BNs in which each CPT is specified by an r -symmetric function for some fixed r .

2.4 OTHER RELATED WORK

Motivated by the practical importance of inference problems (Darwiche (2009); Koller and Friedman (2009); Pearl (1988)), research in this area has proceeded along two primary directions. The first direction focuses on the development of efficient heuristics that can be used to obtain fast solutions to problems that arise in practice (see for example (Chavira, 2007; Dechter, 1999) and the references cited therein). The second direction is the identification of restricted versions of inference problems that can be solved efficiently. As mentioned earlier, an important step in that direction is the work of Lauritzen and Spiegelhalter (1988) which provides an efficient algorithm for inference problems for treewidth bounded (moralized) BNs. Other references that consider inference problems for restricted versions of BNs include (Bacchus et al., 2003; Boutilier et al., 1996; Dechter, 1999; Jensen et al., 1990). The notion of causal independence used in Zhang and Poole (1996) relies on conditional probability tables that are essentially symmetric functions. We note that symmetric functions have also been used in the context of lifted inference (Jha et al., 2010; Milch et al., 2008).

Another approach, called **parent divorcing**, for dealing with nodes of large indegrees was introduced in Olesen et al. (1989). The basic idea of this approach is to modify a given BN in the following manner: when a node has a large indegree, the subgraph consisting of the node and its predecessors is replaced by a directed tree in which each node

has a small indegree. An example to illustrate this approach is shown in Figure 2. There are two main difficulties with this approach. The first is that the treewidth of the resulting BN can be much larger than that of the given BN. The second difficulty is that the size of domain from which newly added nodes take on values may become large. These two aspects can significantly increase the running time of the algorithms for the inference problems. Our algorithms can handle nodes with unbounded indegrees without modifying the given BN, provided all the CPTs are described by r -symmetric functions for some fixed integer r .

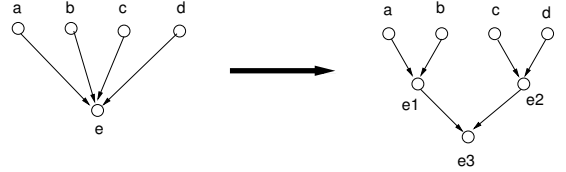


Figure 2: An Example for Parent Divorcing Approach

3 POLYNOMIAL TIME ALGORITHMS FOR TREewidth-BOUNDED BNs WITH SYMMETRIC CPTs

3.1 OVERVIEW

This section presents polynomial time algorithms for inference problems for treewidth-bounded BNs where each probability table is represented as symmetric function. We assume that a BN is given along with its tree decomposition of treewidth k , for some fixed integer $k \geq 1$. We will present the details of the algorithm for the PROB problem. The modifications needed to solve the INF and MPE problems and the extension of the algorithm to handle r -symmetric CPTs for any fixed $r \geq 1$ are presented in Rosenkrantz et al. (2014).

3.2 NOTES ON TREE DECOMPOSITION

This section mentions some known facts about tree decompositions and also reviews some related terminology.

We assume that one of the nodes of the tree decomposition is selected as the root so that the tree decomposition can be viewed as a rooted tree. When a graph G has bounded treewidth, it is well known that a tree decomposition $(\{X_i \mid i \in I\}, T = (I, F))$ of G can be constructed in time that is a polynomial in the size of G . Moreover, this can be done so that all of the following conditions hold (Barrett et al., 2007a,b; Bodlaender, 1997): (a) T is a binary tree; that is, each node of T has at most two children. (b) The number of nodes of T with fewer than two children is $\leq n$, the number of nodes in G . (c) The number of nodes of T with two children is $\leq n$. Our algorithm relies on this special form of tree decomposition.

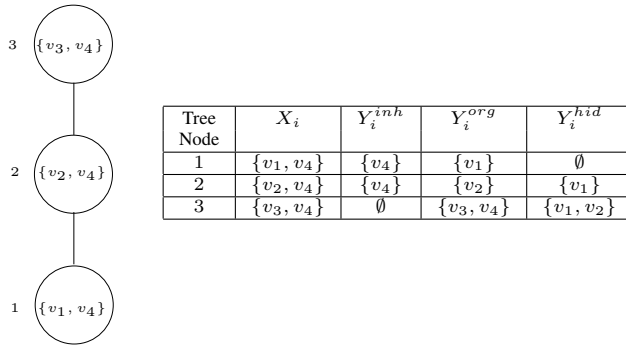


Figure 3: A Tree Decomposition for the BN shown in Figure 1. For each tree node i , X_i , Y_i^{inh} , Y_i^{org} and Y_i^{hid} denote respectively the set of explicit, inherited, originating and hidden nodes respectively, $1 \leq i \leq 3$.

The following terminology regarding nodes in tree decompositions is from (Barrett et al., 2007a,b). Let T be the given tree decomposition of a BN G . For a given node i of T , the nodes of G in X_i are called **explicit nodes** of i . If a given explicit node v of i is also an explicit node of the parent of i , then v is referred to as an **inherited node** of i ; and if v does not occur in the parent of i , then v is called an **originating node** of i .

We refer to the set of all explicit nodes occurring in the subtree of T rooted at i that are not explicit nodes of i as **hidden nodes** of i . (Thus, the hidden nodes of i are the union of the originating and hidden nodes of the children of i .) For any node i in T , we use Y_i^{inh} , Y_i^{org} and Y_i^{hid} to denote respectively the set of inherited nodes, the set of originating nodes and the set of hidden nodes of i .

Example: A tree decomposition for the BN of Figure 1 is shown in Figure 3. The tree decomposition has three nodes. For each tree node, the set of explicit nodes is shown. The table in Figure 3 also shows the explicit, inherited, originating and hidden sets for each tree node. \square

3.3 CONFIGURATIONS AND SIGNATURES

For all of the computational problems we consider, we are given a BN $G(V, E)$ and an observation c_O . As mentioned earlier, we also assume that each CPT is specified as a symmetric function.

Definition 3.1 (a) Let Y be a set of nodes of the given BN G . We refer to a partial configuration on Y as a **Y -configuration**.

(b) Let Y and W be not necessarily disjoint sets of nodes of the given BN G . We say that a given Y -configuration α and a given W -configuration β are **consistent** if for all nodes z in $Y \cap W$, $Y(z) = W(z)$.

(c) Given the observation (i.e., O -configuration) c_O , we

say that a given Y -configuration α is **valid** if for all nodes w assigned values in both c_O and α , $c_O(w) = \alpha(w)$, i.e., α and c_O are consistent.

(d) Γ_Y denotes the set of all valid Y -configurations.

(e) Let Y and W be sets of nodes of the given BN G such that $W \subseteq Y$. Let α be a Y -configuration. We define the **restriction** of α to W , denoted as $\alpha|W$, to be the W -configuration obtained by restricting α to the members of W .

The concept of a signature, defined below for the case where each CPT is a symmetric function, plays an important role in our algorithm.

Definition 3.2 Let Y and W be not necessarily disjoint sets of nodes of the given BN G . (a) Let α be a Y -configuration. The **signature** of α with respect to W , denoted as $\text{sig}(\alpha, W)$, specifies for each $w \in W$, the number of parents of w that are set to 1 by α . We refer to such a signature as a (Y, W) -signature.

(b) Suppose Γ is a set of Y -configurations. The **signature** of Γ with respect to W is the union of the signature of each $\gamma \in \Gamma$ with respect to W .

(c) We say that a given (Y, W) -signature is **valid** if it is $\text{sig}(\alpha, W)$ for some valid Y -configuration α .

(d) $H_{Y,W}$ denotes the set of all valid (Y, W) -signatures.

Example: Consider the BN shown in Figure 1. Let $Y = \{v_2, v_3\}$ and $W = \{v_1, v_4\}$. Consider the Y -configuration γ which sets $v_2 = 0$ and $v_3 = 1$. It is easy to see that γ sets 0 of v_1 's parents to 1 and exactly one of v_4 's parents to 1. So $\text{sig}(\gamma, W)$, the signature of γ with respect to W , can be represented as $[v_1 : 0, v_4 : 1]$. Suppose that the given observation c_O does not specify the value of any node. Then Γ_Y contains four Y -configurations. By computing the union of the signatures of these four Y -configurations, it can be seen that $H_{Y,W}$ is the set $\{[v_1 : 0, v_4 : 0], [v_1 : 0, v_4 : 1], [v_1 : 0, v_4 : 2]\}$. \square

If σ is (Y, W) -signature, then for any $w \in W$, we use $\sigma(w)$ to denote the value specified by σ for w . Using this notation, we define some operations on signatures which produce new signatures. These operations are used by our algorithm.

Definition 3.3 Let G be a BN and let W be a subset of nodes of G .

(a) Let σ and σ' be two signatures with respect to a node set W . The **sum** of the two signatures is another signature denoted by $\sigma + \sigma'$, such that for each $w \in W$, $(\sigma + \sigma')(w) = \sigma(w) + \sigma'(w)$.

(b) Let σ be a signature with respect to a node set W and let Y be a subset of W . The **restriction** of σ to Y is another signature denoted by $\sigma|Y$, such that for each $y \in Y$, $(\sigma|Y)(y) = \sigma(y)$.

(c) Let σ be a signature with respect to a node set W and let X be a superset of W . The **extension** of σ to X is another signature denoted by $\text{ext}(g, X)$, which is defined as follows: for each $x \in X$, if $x \in W$, then $\text{ext}(\sigma, X)(x) = \sigma(x)$; otherwise, $\text{ext}(\sigma, X)(x) = 0$.

Suppose w is a node of G and η is a partial configuration that specifies a value (namely, $\eta(w)$) for w and for every parent of w . Given these values, the CPT for w specifies a probability value for $\eta(w)$, which will be denoted by $p_\eta(w)$. Suppose W is a subset of nodes of G and η is a partial configuration that specifies a value for every node $w \in W$ and for every parent of every node $w \in W$. Thus, for every node $w \in W$, the value $p_\eta(w)$ is defined. We define $p_\eta(W)$ by

$$p_\eta(W) = \prod_{w \in W} p_\eta(w). \quad (1)$$

We also need a slight extension of the definition given by Equation (1). Let X and Y be disjoint sets of nodes of G . Let η be a X -configuration, and σ be a (Y, X) -signature. Suppose that for a given node w in X , all the parents of w are in $X \cup Y$. Because the CPT for w is given by a symmetric function, given the values that η assigns to those parents of w that are in X , and the value that σ assigns to w , the CPT for w assigns a probability value to $\eta(w)$. We denote this probability value as $p_{\eta, \sigma}(w)$. Suppose that W is a subset of X , such that for every node w in W , all the parents of w are in $X \cup Y$. (Thus, for every node w in W , the value $p_{\eta, \sigma}(w)$ is defined.) Now, we define $p_{\eta, \sigma}(W)$ by

$$p_{\eta, \sigma}(W) = \prod_{w \in W} p_{\eta, \sigma}(w). \quad (2)$$

3.4 ALGORITHM FOR THE PROB PROBLEM

Recall that in the PROB problem, we are given a BN $G(V, E)$ and an observation c_O . The goal is to find the probability of c_O , that is, the sum of the probabilities of all (complete) configurations that are extensions of c_O . Let the constant k denote the treewidth of G . We assume that a tree decomposition $(\{X_i \mid i \in I\}, T = (I, F))$ of G satisfying all the conditions mentioned in Section 3.2 is also given and that each CPT is specified as a symmetric function.

3.4.1 Information Maintained by the Algorithm

Our algorithm solves the PROB problem for G by using bottom-up dynamic programming on the tree decomposition T . The algorithm maintains information for each node of T , as summarized in Table 1, and described below.

For each node i of T , the algorithm maintains the two sets of signatures $H_{Y_i^{hid}, X_i}$ and $H_{(Y_i^{hid} \cup Y_i^{org}), Y_i^{inh}}$, plus two tables of probability values, which we denote as Q^i and R^i . We now provide a description of these signature sets and tables for each tree node i .

- (a) $H_{Y_i^{hid}, X_i}$ is the set of all valid (Y_i^{hid}, X_i) -signatures. (Recall that Y_i^{hid} is the set of hidden nodes of i , and X_i is the set of explicit nodes of i .)
- (b) $H_{(Y_i^{hid} \cup Y_i^{org}), Y_i^{inh}}$ is the set of all valid $(Y_i^{hid} \cup Y_i^{org}, Y_i^{inh})$ -signatures. (Recall that Y_i^{inh} is the set of inherited nodes of i .)
- (c) Table Q^i contains a probability value for each pair in $\Gamma_{X_i} \times H_{Y_i^{hid}, X_i}$.

Consider a given element of table Q^i , say $Q^i[\alpha, \sigma]$, where α is a valid X_i -configuration and σ is a valid (Y_i^{hid}, X_i) -signature. The value of $Q^i[\alpha, \sigma]$ is defined by

$$Q^i[\alpha, \sigma] = \sum_{\beta} p_{\alpha \cup \beta}(Y_i^{hid}) \quad (3)$$

where the summation is over all β such that β is a valid Y_i^{hid} -configuration and $\text{sig}(\beta, X_i) = \sigma$.

Note that the definition of a tree decomposition ensures that every parent of a hidden node of i is either an explicit node or a hidden node of i , so each probability value occurring in Equation (3) is well defined.

- (d) Table R^i contains an entry for each pair in $\Gamma_{Y_i^{inh}} \times H_{(Y_i^{hid} \cup Y_i^{org}), Y_i^{inh}}$.

Consider a given element of table R^i , say $R^i[\psi, \theta]$, where ψ is a valid Y_i^{inh} -configuration and θ is a valid $(Y_i^{hid} \cup Y_i^{org}, Y_i^{inh})$ -signature. The value of $R^i[\psi, \theta]$ is defined by

$$R^i[\psi, \theta] = \sum_{\beta} p_{\psi \cup \beta}(Y_i^{hid} \cup Y_i^{org}) \quad (4)$$

where the summation is over all β such that β is a valid $(Y_i^{hid} \cup Y_i^{org})$ -configuration and $\text{sig}(\beta, Y_i^{inh}) = \theta$.

Note that the definition of a tree decomposition ensures that every parent of an hidden or originating node of i is either an explicit node or a hidden node of i , so each probability value occurring in Equation (4) is well defined.

Equation (3) represents the definition of each entry of Q^i . However, for a given α and σ , one *cannot* use the equation directly to efficiently compute the value of $Q^i[\alpha, \sigma]$, since the number of valid configurations to be considered may be exponential in the number of nodes of G . A similar comment applies to the computation of $R^i[\psi, \sigma]$ directly using Equation (4). How these values can be computed efficiently is discussed below.

3.4.2 Description of the Algorithm

Having described the information maintained by the algorithm, we can now describe the the bottom-up construction

Table 1: Notation used in Describing the Dynamic Programming Algorithm for the PROB Problem

Symbol	Explanation
X_i	The set of explicit nodes of tree node i .
Y_i^{inh}	The set of inherited nodes of tree node i .
Y_i^{org}	The set of originating nodes of tree node i .
Y_i^{hid}	The set of hidden nodes of tree node i .
Γ_{X_i}	The set of valid partial configurations on the explicit nodes of i
$\Gamma_{Y_i^{inh}}$	The set of valid partial configurations on the inherited nodes of i
$H_{Y_i^{hid}, X_i}$	The set of signatures of all valid partial configurations of the hidden nodes of i with respect to the explicit nodes of i .
$H_{(Y_i^{hid} \cup Y_i^{org}), Y_i^{inh}}$	The set of signatures of all valid partial configurations of the hidden and originating nodes of i with respect to the inherited nodes of i .
$Q^i[\Gamma_{X_i}, H_{Y_i^{hid}, X_i}]$	$Q^i[\alpha, \sigma]$ maps valid partial configuration α on the explicit nodes of i and signature $\sigma \in H_{Y_i^{hid}, X_i}$ to a probability value.
$R^i[\Gamma_{Y_i^{inh}}, H_{(Y_i^{hid} \cup Y_i^{org}), Y_i^{inh}}]$	$R^i[\psi, \theta]$ maps valid partial configuration ψ on the inherited nodes of i and signature $\theta \in H_{(Y_i^{hid} \cup Y_i^{org}), Y_i^{inh}}$ to a probability value.

of the signature sets and tables for each node of the tree decomposition. We present the construction in the following order.

1. First, we describe the computation of the set $H_{Y_i^{hid}, X_i}$ and the Q^i table for a *leaf* node i of the tree decomposition.
2. Next, we describe the computation of set $H_{(Y_i^{hid} \cup Y_i^{org}), Y_i^{inh}}$ and the R^i table for an *arbitrary* node i of the tree decomposition, given set $H_{Y_i^{hid}, X_i}$ and table Q^i .
3. Then we describe the computation of set $H_{Y_i^{hid}, X_i}$ and table Q^i for a *nonleaf* node i of the tree decomposition, given the $H_{(Y_i^{org} \cup Y_i^{hid}), Y_i^{inh}}$ sets and R tables for the children of node i in the tree decomposition.
4. Finally, we indicate how the solution for the PROB problem can be computed from the values computed for the root of the tree decomposition.

We now present the details for each of the four parts above. The operations on signatures defined in Section 3.3 are used in the following description.

Part 1: Consider a leaf node i of the tree decomposition. Note that leaf node i contains no hidden nodes. Consequently, $H_{Y_i^{hid}, X_i}$ consists of a single signature σ , which maps each node of X_i into the value 0.

For each valid X_i -configuration α , the table entry $Q^i[\alpha, \sigma]$ is given the value 1. Pseudocode for Part 1 is presented in Figure 4.

Part 2: For any node i of the tree decomposition, given set $H_{Y_i^{hid}, X_i}$ and table Q^i , set $H_{(Y_i^{hid} \cup Y_i^{org}), Y_i^{inh}}$ and table R^i can be constructed as follows.

1. $H_{Y_i^{hid}, X_i} = \{\sigma\}$, where σ is the signature that maps each node $x \in X_i$ into the value 0.
2. For each valid partial configuration α on X_i , $Q^i[\alpha, \sigma] = 1$.

Figure 4: Pseudocode for Part 1 of the Algorithm for the PROB Problem

Recall that $\Gamma_{Y_i^{org}}$ denotes the set of all valid Y_i^{org} -configurations, and that for any $\gamma \in \Gamma_{Y_i^{org}}$, $sig(\gamma, Y_i^{inh})$ denotes the signature of γ with respect to Y_i^{inh} .

Computation of $H_{(Y_i^{hid} \cup Y_i^{org}), Y_i^{inh}}$: This quantity is computed using the following equation

$$H_{(Y_i^{hid} \cup Y_i^{org}), Y_i^{inh}} = \bigcup_{\gamma, \sigma'} sig(\gamma, Y_i^{inh}) + (\sigma' | Y_i^{inh})$$

where the union is over each pair γ, σ' such that $\gamma \in \Gamma_{Y_i^{org}}$ and $\sigma' \in H_{Y_i^{hid}, X_i}$.

Computation of R^i : Consider an entry in the Q^i table, say $Q^i[\alpha, \sigma]$. The valid X_i -configuration α can be considered to be the disjoint union of the valid Y_i^{inh} -configuration $\psi = \alpha | Y_i^{inh}$ and the valid Y_i^{org} -configuration $\gamma = \alpha | Y_i^{org}$. Similarly, the (Y_i^{hid}, X_i) -signature σ can be considered to be the disjoint union of the (Y_i^{hid}, Y_i^{inh}) -signature $\sigma' = \sigma | Y_i^{inh}$ and the (Y_i^{hid}, Y_i^{org}) -signature $\sigma'' = \sigma | Y_i^{org}$. Let θ be the $(Y_i^{hid} \cup Y_i^{org}, Y_i^{inh})$ -signature $\sigma' + sig(\gamma, Y_i^{inh})$. The entry $Q^i[\alpha, \sigma]$ of the Q^i table contributes to the value of the entry $R^i[\psi, \theta]$ of the R^i table. The value of this contribution is the product $Q^i[\alpha, \sigma] * p_{\alpha, \sigma}(Y_i^{org})$.

The value $R^i[\psi, \theta]$ can be computed using the following equation.

$$R^i[\psi, \theta] = \sum Q^i[\alpha, \sigma] * p_{\alpha, \sigma}(Y_i^{org})$$

where the summation is over each $\alpha \in \Gamma_{X_i}$ and $\sigma \in H_{Y_i^{hid}, X_i}$ such that $(\psi = \alpha | Y_i^{inh}) \wedge (\theta = (\sigma | Y_i^{inh}) + sig(\alpha | Y_i^{org}, Y_i^{inh}))$.

Alternatively, the R^i table can be computed by first setting all the entries in the table to zero, and then scanning the Q^i table, adding the contribution of each entry in the Q^i table to the appropriate entry in the R^i table. Pseudocode for Part 2, using this approach to computing the R^i table, is shown in Figure 5.

Computation of $H_{(Y_i^{hid} \cup Y_i^{org}), Y_i^{inh}}$:

1. Initialization: $H_{(Y_i^{hid} \cup Y_i^{org}), Y_i^{inh}} = \emptyset$.
2. **for** each valid Y_i^{org} -configuration γ **do**
 - (i) Compute $\sigma' = sig(\gamma, Y_i^{inh})$, the signature of γ with respect to Y_i^{inh} .
 - (ii) **for** each signature $\sigma \in H_{Y_i^{hid}, X_i}$ **do**
 - (a) $\sigma'' = \sigma' + (\sigma | Y_i^{inh})$.
 - (b) $H_{(Y_i^{hid} \cup Y_i^{org}), Y_i^{inh}} = H_{(Y_i^{hid} \cup Y_i^{org}), Y_i^{inh}} \cup \{\sigma''\}$.

Computation of R^i :

- for** each valid Y_i^{inh} -configuration ψ **do**
for each signature $\theta \in H_{(Y_i^{hid} \cup Y_i^{org}), Y_i^{inh}}$ **do**
 $R^i[\psi, \theta] = 0$.
- for** each valid X_i -configuration α **do**
 1. $\psi = \alpha | Y_i^{inh}$.
 2. **for** each signature $\sigma \in H_{Y_i^{hid}, X_i}$ **do**
 - (a) Compute $\theta' = sig(\alpha | Y_i^{org}, Y_i^{inh})$, the signature of $\alpha | Y_i^{org}$ with respect to Y_i^{inh} .
 - (b) $\theta = (\sigma | Y_i^{inh}) + \theta'$.
 - (c) $R^i[\psi, \theta] = R^i[\psi, \theta] + Q^i[\alpha, \sigma] * p_{\alpha, \sigma}(Y_i^{org})$.

Figure 5: Pseudocode for Part 2 of the Algorithm for the PROB Problem

Part 3: We now consider computing set $H_{Y_i^{hid}, X_i}$ and table Q^i for a nonleaf node i of the tree decomposition.

Case 1: Nonleaf node i has only one child.

Let i_1 denote the child of i in the tree decomposition. We compute $H_{Y_i^{hid}, X_i}$ as

$$H_{Y_i^{hid}, X_i} = \{ext(\theta_1, X_i) \mid \theta_1 \text{ is in } H_{(Y_{i_1}^{hid} \cup Y_{i_1}^{org}), Y_{i_1}^{inh}}\}.$$

Given the table R^{i_1} for i_1 , the table Q^i is constructed as follows. Consider a given entry $Q^i[\alpha, \sigma]$, for X_i -

configuration α and (Y_i^{hid}, X_i) -signature σ . The value of this entry is set to the value of $R^{i_1}[\alpha | Y_{i_1}^{inh}, \sigma | Y_{i_1}^{inh}]$.

Case 2: Nonleaf node i has two children.

Let i_1 and i_2 denote the children of i in the tree decomposition. We compute $H_{Y_i^{hid}, X_i}$ as

$$H_{Y_i^{hid}, X_i} = \bigcup_{\theta_1, \theta_2} ext(\theta_1, X_i) + ext(\theta_2, X_i).$$

where the union is over all pairs θ_1 and θ_2 such that $\theta_1 \in H_{(Y_{i_1}^{hid} \cup Y_{i_1}^{org}), Y_{i_1}^{inh}}$ and $\theta_2 \in H_{(Y_{i_2}^{hid} \cup Y_{i_2}^{org}), Y_{i_2}^{inh}}$.

The tables R^{i_1} and R^{i_2} for tree nodes i_1 and i_2 are combined to produce table Q^i for tree node i as follows. For any $\theta_1 \in H_{(Y_{i_1}^{hid} \cup Y_{i_1}^{org}), Y_{i_1}^{inh}}$ and $\theta_2 \in H_{(Y_{i_2}^{hid} \cup Y_{i_2}^{org}), Y_{i_2}^{inh}}$, let

$$\sigma = ext(\theta_1, X_i) + ext(\theta_2, X_i).$$

For any valid X_i -configuration α , the table entries $R^{i_1}[\alpha | Y_{i_1}^{inh}, \theta_1]$ and $R^{i_2}[\alpha | Y_{i_2}^{inh}, \theta_2]$ together contribute to the value of $Q^i[\alpha, \sigma]$. The value of this contribution is $R^{i_1}[\alpha | Y_{i_1}^{inh}, \theta_1] * R^{i_2}[\alpha | Y_{i_2}^{inh}, \theta_2]$.

Consider a given a valid X_i -configuration α and signature σ in $H_{Y_i^{hid}, X_i}$. We can compute $Q^i[\alpha, \sigma]$ as a sum of products:

$$Q^i[\alpha, \sigma] = \sum_{\theta_1, \theta_2} R^{i_1}[\alpha | Y_{i_1}^{inh}, \theta_1] * R^{i_2}[\alpha | Y_{i_2}^{inh}, \theta_2]$$

where the summation is over all pairs θ_1 and θ_2 such that $\theta_1 \in H_{(Y_{i_1}^{hid} \cup Y_{i_1}^{org}), Y_{i_1}^{inh}}$, $\theta_2 \in H_{(Y_{i_2}^{hid} \cup Y_{i_2}^{org}), Y_{i_2}^{inh}}$ and $\sigma = ext(\theta_1, X_i) + ext(\theta_2, X_i)$.

Alternatively, the Q^i table can be computed by first setting all the entries in the table to zero, and then scanning the R^{i_1} and R^{i_2} tables, adding the contribution of each pair of entries in these tables to the appropriate entries in the R^i table. Pseudocode for Part 3, using this approach to computing the Q^i table, is shown in Figure 6.

Part 4: Let r be the root node of the tree decomposition. The root node has no inherited nodes, so $Y_r^{inh} = \emptyset$. Consequently, $\Gamma_{Y_r^{inh}}$ contains only the empty partial configuration, which we denote as ψ_\emptyset . Also, set $H_{(Y_r^{org} \cup Y_r^{hid}), Y_r^{inh}}$ contains only the empty signature, which we denote as σ_\emptyset . Table R^r consists of a single entry, $R^r[\psi_\emptyset, \sigma_\emptyset]$. The value of $R^r[\psi_\emptyset, \sigma_\emptyset]$ is the solution to the PROB problem.

3.4.3 Running Time Analysis

We now state a result that gives the running time of the algorithm presented in the previous section. A proof of this result appears in Rosenkrantz et al. (2014).

Lemma 3.4 *The dynamic programming algorithm for the PROB problem runs in $O(n^{2k+3})$ time, where n is the number of nodes in the given Bayesian network G and k is the treewidth of G .*

Case 1: Node i has only one child i_1 in the tree decomposition.

Computation of $H_{Y_i^{hid}, X_i}$:

1. Initialization: $H_{Y_i^{hid}, X_i} = \emptyset$.
2. **for** each signature $\theta_1 \in H_{(Y_{i_1}^{hid} \cup Y_{i_1}^{org}), Y_{i_1}^{inh}}$ **do**
 $H_{Y_i^{hid}, X_i} = H_{Y_i^{hid}, X_i} \cup \{ext(\theta_1, X_i)\}.$

Computation of Q^i :

for each valid X_i -configuration α **do**
for each signature $\sigma \in H_{Y_i^{hid}, X_i}$ **do**
 $Q^i[\alpha, \sigma] = R^{i_1}[\alpha | Y_{i_1}^{inh}, \sigma | Y_{i_1}^{inh}].$

Case 2: Node i has two children i_1 and i_2 in the tree decomposition.

Computation of $H_{Y_i^{hid}, X_i}$:

1. Initialization: $H_{Y_i^{hid}, X_i} = \emptyset$.
2. **for** each signature $\theta_1 \in H_{(Y_{i_1}^{hid} \cup Y_{i_1}^{org}), Y_{i_1}^{inh}}$ **do**
for each signature $\theta_2 \in H_{(Y_{i_2}^{hid} \cup Y_{i_2}^{org}), Y_{i_2}^{inh}}$ **do**
 - (a) $\sigma = ext(\theta_1, X_i) + ext(\theta_2, X_i).$
 - (b) $H_{Y_i^{hid}, X_i} = H_{Y_i^{hid}, X_i} \cup \{\sigma\}.$

Computation of Q^i :

for each valid X_i -configuration α **do**
for each signature $\sigma \in H_{Y_i^{hid}, X_i}$ **do**
 $Q^i[\alpha, \sigma] = 0.$

for each signature $\theta_1 \in H_{(Y_{i_1}^{hid} \cup Y_{i_1}^{org}), Y_{i_1}^{inh}}$ **do**
for each signature $\theta_2 \in H_{(Y_{i_2}^{hid} \cup Y_{i_2}^{org}), Y_{i_2}^{inh}}$ **do**

- (a) $\sigma = ext(\theta_1, X_i) + ext(\theta_2, X_i).$
- (b) **for** each valid X_i -configuration α **do**
 $Q^i[\alpha, \sigma] = Q^i[\alpha, \sigma] + R^{i_1}[\alpha | Y_{i_1}^{inh}, \theta_1] * R^{i_2}[\alpha | Y_{i_2}^{inh}, \theta_2].$

Figure 6: Pseudocode for Part 3 of the Algorithm for the PROB Problem

Since k is fixed, our algorithm for the PROB problem runs in polynomial time. Thus, the following theorem summarizes the main result of Section 3.4.

Theorem 3.5 *The PROB problem can be solved efficiently for the class of treewidth-bounded BNs where each CPT is specified as a symmetric function.* ■

4 HARDNESS RESULTS

The results in the previous section show that inference problems for treewidth bounded BNs can be solved efficiently even when the indegrees of nodes are not bounded, provided each CPT is expressed as a symmetric function. The following result points out the tightness of these results; in particular, the result shows that the problems remain computationally intractable even if one of the conditions is violated. A proof of this result appears in Rosenkrantz et al. (2014).

Proposition 4.1 (a) *If CPTs are not required to be r -symmetric, then the PROB problem is #P-hard even when the BN is a directed tree (whose treewidth is 1).*

(b) *When the treewidth of the BN is not bounded, the PROB problem is #P-hard even when the CPT at each node is given by a symmetric function.*

5 CONCLUSIONS

We presented efficient algorithms for exact inference problems for BNs when the unmoralized graph is treewidth-bounded and each CPT is an r -symmetric function for a fixed r . We also observed that if either of these conditions is relaxed, the inference problems are computationally intractable.

We conclude by mentioning two general directions for further research. First, dynamic programming algorithms for treewidth-bounded BNs require memory that grows exponentially with the treewidth. It will be useful to develop practical techniques that can significantly reduce the amount of memory needed. Second, it is of interest to identify additional restrictions on BNs (based on problem instances that arise in practice) that can lead to practical algorithms for large problem instances.

Acknowledgments

We thank the UAI 2014 reviewers for providing helpful comments. This work has been partially supported by the following grants: DTRA Grant HDTRA1-11-1-0016, DTRA CNIMS Contract HDTRA1-11-D-0016-0010, NSF Career CNS 0845700, NSF ICES CCF-1216000, NSF NETSE Grant CNS-1011769 and DOE DE-SC0003957.

References

- A. M. Abdelbar, S. T. Hedetniemi, and S. M. Hedetniemi. The complexity of approximating MAPs for belief networks with bounded probabilities. *Artificial Intelligence*, 124(2):283–288, 2000.
- F. Bacchus, S. Dalmao, and T. Pitassi. Algorithms and complexity results for #SAT and Bayesian inference. In *Proc. FOCS*, pages 340–351, 2003.
- C. L. Barrett, H. B. Hunt III, M. V. Marathe, S. S. Ravi, D. J. Rosenkrantz, R. E. Stearns, and M. Thakur. Computational aspects of analyzing social network dynamics. *Proc. IJCAI*, pages 2268–2273, 2007a.
- C. L. Barrett, H. B. Hunt III, M. V. Marathe, S. S. Ravi, D. J. Rosenkrantz, R. E. Stearns, and M. Thakur. Predecessor existence problems for finite discrete dynamical systems. *Theoretical Computer Science*, 386:3–37, Oct. 2007b.
- H. Bodlaender. Treewidth: Algorithmic techniques and results. *Proc. 22nd Symposium on Mathematical Foundations of Computer Science*, pages 29–36, 1997.
- H. Bodlaender. Probabilistic networks: Inference and other problems. Lecture Slides – Institute for Programming Research and Algorithmics, 2004.
- H. L. Bodlaender. A tourist guide through treewidth. *Acta Cybernetica*, 11(1-2):1–22, 1993.
- C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-specific independence in Bayesian networks. In *Proc. UAI*, pages 115–123, 1996.
- M. Chavira. *Beyond Treewidth in Probabilistic Inference*. PhD thesis, Computer Science Dept., UCLA, 2007.
- G. F. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42:393–405, 1990.
- B. Courcelle and M. Mosbah. Monadic second-order evaluations on tree-decomposable graphs. *Theoretical Computer Science*, 109:49–82, 1993.
- B. Courcelle, J. Makowsky, and U. Rotics. On the fixed parameter complexity of graph enumeration problems definable in monadic second order logic. *Discrete Applied Mathematics*, 108:23–52, 2001.
- P. Dagum and M. Luby. Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artificial Intelligence*, 60:141–153, 1993.
- A. Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, New York, NY, 2009.
- C. P. de Campos. New complexity results for MAP in Bayesian networks. In *Proc. IJCAI*, pages 2100–2106, 2011.
- R. Dechter. Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, 113:41–85, 1999.
- E. Fischer, S. Gould, and E. Ravve. Counting truth assignments of formulas of bounded tree-width or clique-width. *Discrete Applied Mathematics*, 156:511–529, 2008.
- G. Gottlob and S. Szeider. Fixed-parameter algorithms for artificial intelligence, constraint satisfaction and database problems. *The Computer Journal*, 51(3):303–325, 2008.
- R. Impagliazzo and R. Paturi. On the complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.
- F. Jensen, S. Lauritzen, and K. Olesen. Bayesian updating in causal probabilistic networks by local computation. *Computational Statistics Quarterly*, 4:269–282, 1990.
- A. K. Jha, V. Gogate, A. Meliou, and D. Suciu. Lifted inference seen from the other side: The tractable features. In *Proc. NIPS*, pages 973–981, 2010.
- D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, Cambridge, MA, 2009.
- J. Kwisthout, H. L. Bodlaender, and L. C. van der Gaag. The necessity of bounded treewidth for efficient inference in Bayesian networks. In *Proc. ECAI*, pages 237–242, 2010.
- S. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their applications to expert systems. *J. Royal Statistical Society, Series B*, 50(2):157–224, 1988.
- B. Milch, L. S. Zettlemoyer, K. Kresting, M. Haimes, and L. P. Kaelbling. Lifted probabilistic inference with counting formulas. In *Proc. AAAI*, pages 1062–1068, 2008.
- K. G. Olesen, U. Kjaerulff, F. Jensen, F. V. Jensen, B. Falck, S. Andreassen, and S. K. Andersen. A MUNIN network for the median nerve – A case study on loops. *Applied Artificial Intelligence*, 3(2-3):385–403, Oct. 1989.
- J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan-Kaufmann Publishers, San Mateo, CA, 1988.
- N. Robertson and P. Seymour. Graph minors II: Algorithmic aspects of treewidth. *J. Algorithms*, 7:309–322, 1986.
- D. J. Rosenkrantz, A. K. Vullikanti, M. V. Marathe, and S. S. Ravi. Bayesian inference in treewidth-bounded graphical models without indegree constraints. Technical Report 14-066, NDSSL, VBI, Blacksburg, VA, June 2014.
- M. Samer and S. Szeider. Algorithms for propositional model counting. In *Proc. LPAR*, pages 484–498, 2007.
- N. Zhang and D. Poole. Exploiting causal independence in bayesian network inference. *Journal of Artificial Intelligence Research*, 5:301–328, 1996.

SPPM: Sparse Privacy Preserving Mappings

Salman Salamatian ^{*} Nadia Fawaz [†] Branislav Kveton [†] Nina Taft [†]

^{*} École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland

[†] Technicolor, USA

Abstract

We study the problem of a user who has both public and private data, and wants to release the public data, e.g. to a recommendation service, yet simultaneously wants to protect his private data from being inferred via big data analytics. This problem has previously been formulated as a convex optimization problem with linear constraints where the objective is to minimize the mutual information between the private and released data. This attractive formulation faces a challenge in practice because when the underlying alphabet of the user profile is large, there are too many potential ways to distort the original profile. We address this fundamental scalability challenge. We propose to generate sparse privacy-preserving mappings by recasting the problem as a sequence of linear programs and solving each of these incrementally using an adaptation of Dantzig-Wolfe decomposition. We evaluate our approach on several datasets and demonstrate that nearly optimal privacy-preserving mappings can be learned quickly even at scale.

1 Introduction

Finding the right balance between privacy risks and big data rewards is one of the biggest challenges facing society today. Big data creates tremendous opportunity, especially for all of the services that offer personalized advice. Recommendation services are rampant today and offer advice on everything including movies, TV shows, restaurants, music, sleep, exercise, vacation, entertainment, shopping, and even friends. On one hand, people are willing to part with some of their personal data (e.g. movie watching history) for the sake of these services. On the other hand, many users have some data about themselves they would prefer to keep private (e.g. their political affiliation, salary, pregnancy status, religion). Most individuals have both public and private data and hence they need to maintain a boundary between these different ele-

ments of their personal information. This is an enormous challenge because inference analysis on publicly released data can often uncover private data [27, 6, 21].

A number of research efforts have explored the idea of distorting the released data [29, 26, 17, 7, 11, 3] to preserve user’s privacy. In some of these prior efforts, distortion aims at creating some confusion around user data by making its value hard to distinguish from other possible values; in other efforts, distortion is designed to counter a particular inference threat (i.e. a specific classifier or analysis). Recently [7] proposed a new framework for data distortion, based on information theory, that captures privacy leakage in terms of mutual information. Minimizing the mutual information between a user’s private data and released data is attractive because it reduces the correlation between the private data and the publicly released data, and thus any inference analysis that tries to learn the private data from the public data is rendered weak, if not useless. In other words, this approach is agnostic to the type of inference analysis used in a given threat.

This promising framework, while theoretically sound, faces some challenges in terms of bridging the gap between theory and practicality. Distorting data, in the context of recommendation systems, means altering a user’s profile. The framework casts this privacy problem as a convex optimization problem with linear constraints, where the number of variables grows quadratically with the size of the underlying alphabet that describes user’s profiles. In real world systems, the alphabet can be huge, thus the enormous number of options for distorting user profiles presents a scalability challenge, that we address in this paper.

We make two contributions to handle scalability. First, by studying small scale problems, both analytically and empirically, we identify that mappings to distort profiles are in fact naturally sparse. We leverage this observation to develop sparse privacy-preserving mappings (SPPM). Second, we propose an algorithm, called SPPM that handles scalability through two insights. Although the underlying optimization problem has linear constraints, its objective function is non-linear. We use the Frank-Wolfe algorithm that approximates the objective via a sequence of linear approximations that can be solved quickly. To do this, we

adapt the Dantzig-Wolfe decomposition to the structure of our problem. Overall we reduce the number of variables from quadratic to linear in the number of user profiles. To the best of our knowledge, this work is the first to apply large scale linear programming optimization techniques to privacy problems.

A salient feature of our novel approach is that the distortion applied to each user is personalized, meaning that it is tailored for that user. Our solution does not require applying distortion to profiles in a uniform way, nor requires any monotonic behavior such as mapping a profile to a far neighbor with decreasing likelihood (such as in [19]). We will see that the flexibility our system allows for a better use of the distortion budget where it is needed, and avoids wasting this budget on unnecessary distortions.

Our third contribution is a detailed evaluation on three datasets, in which we compare our solution to an optimal one (when feasible) and to a state-of-the-art solution based on differential privacy (called the Exponential Mechanism [19]). We find that our solutions are close to optimal, and consistently outperform the exponential mechanism (ExpMec) approach in that we achieve more privacy with less distortion. We show that our methods scale well with respect to the number of user profiles and their underlying alphabet.

Related Work. We consider the framework for privacy against statistical inference in [7, 18, 24]. In [24], a method based on quantization was proposed to reduce the number of optimization variables. It was shown that the reduction in complexity does not affect the privacy levels that can be achieved, but comes at the expense of additional distortion. In [18], privacy mappings in the class of parametric additive noise were considered, which allow the number of optimization variables to be reduced to the number of noise parameters. However, this suboptimal solution is not suitable for perfect privacy, as it requires a high distortion. In this paper, we propose to exploit the structure of the optimization to achieve computational speed-ups that will allow scalability. To the best of our knowledge, this is the first paper that evaluates the privacy-utility framework in [7] on such a large scale.

Our use of the information theoretic framework [7] relies on a *local privacy* setting, where users do not trust the entity collecting data, thus each user holds his data locally, and passes it through a privacy-preserving mechanism before releasing it to the untrusted entity. Local privacy dates back to randomized response in surveys [28], and has been considered in privacy for data mining and statistics [2, 12, 20, 16, 4, 7, 18, 9]. Information theoretic privacy metrics have also been considered in [23, 29, 12, 22, 25]. Finally, differential privacy [10] is currently the prevalent notion of privacy in privacy research. In particular, the expo-

nential mechanism, to which we compare our privacy mapping in Section 4.3, was introduced in [19].

The general problem of minimizing a convex function under convex constraints has been studied extensively, and is of crucial importance in many machine learning tasks. The idea of a sparse approximate solutions to those problems has also been studied in the literature and is often called *Sparse Greedy Approximation* [8, 15, 14, 13, 30]. This type of algorithm has been used with success in many applications such as Neural Network [15], Matrix Factorization [14], SVM [13], Boosting [30], etc. We apply this approach to a new problem and adapt it to efficiently handle our scalability challenges. Another common approach to minimizing a convex function is stochastic gradient descent [31]. This approach is preferable when the feasible set has a simple form and is easy to project to. In our case, the feasible set is defined by many constraints, thus we opted for the Frank-Wolfe algorithm.

2 Problem Statement and Challenges

Privacy-Utility Framework: We consider the setting described in [7], where a user has two kinds of data: a vector of personal data $A \in \mathcal{A}$ that he would like to remain private, e.g. his income level, his political views, and a vector of data $B \in \mathcal{B}$ that he is willing to release publicly in order to receive a useful service (such as releasing his media preferences to a recommender service to receive content recommendations). \mathcal{A} and \mathcal{B} are the sets from which A and B can assume values. We assume that the user private attributes A are linked to his data B by the joint probability distribution $p_{A,B}$. Thus, an adversary who would observe B could infer some information about A . To reduce this inference threat, instead of releasing B , the user releases a *distorted version* of B , denoted $\hat{B} \in \hat{\mathcal{B}}$, generated according to a conditional probabilistic mapping $p_{\hat{B}|B}$, called the *privacy-preserving mapping*. Note that the set $\hat{\mathcal{B}}$ may differ from \mathcal{B} . This setting is reminiscent of the *local privacy* setting (e.g. randomized response, input perturbation) [28], where users do not trust the entity collecting data, thus each user holds his data locally, and passes it through a privacy-preserving mechanism before releasing it. The privacy mapping $p_{\hat{B}|B}$ is designed to render any statistical inference of A based on the observation of \hat{B} harder, while preserving some utility to the released data \hat{B} , by limiting the distortion caused by the mapping. Following the framework for privacy-utility against statistical inference in [7], the inference threat is modeled by the mutual information $I(A; \hat{B})$ between the private attributes A and the publicly released data \hat{B} , while the utility requirement is modeled by a constraint on the average distortion $E_{B, \hat{B}}[d(B, \hat{B})] \leq \Delta$, for some distortion metric $d : \mathcal{B} \times \hat{\mathcal{B}} \rightarrow \mathbb{R}^+$, and $\Delta \geq 0$.

In the case of perfect privacy ($I(A; \hat{B}) = 0$), the privacy mapping $p_{\hat{B}|B}$ renders the released data \hat{B} statistically independent from the private data A .

In general, a system that provides privacy protection does not know in advance the inference algorithm that the adversary will run on released data. This framework is appealing because it works regardless of the inference algorithm used by an adversary. In addition, this framework allows for the use of any distortion metric, either generic metrics such as Hamming, l_p , or weighted norms, or specific utility metrics tailored to a given learning algorithm that will run on the released user data. The use of a generic distance in the utility constraint is relevant in several cases. First, the learning algorithm that will run on the released data may not be known in advance to the system providing privacy protection, as it may be proprietary information that belongs to the service provider. Second, the released user data may be used by numerous analyses tasks based on multiple different ML algorithms, in which case the utility constraint should not be limited to one specific distortion metric.

Both the mutual information $I(A; \hat{B})$ and the average distortion $E_{B, \hat{B}}[d(B, \hat{B})]$ depend on both the prior distribution $p_{A,B}$ and the privacy mapping $p_{\hat{B}|B}$, since $A \rightarrow B \rightarrow \hat{B}$ form a Markov chain. To stress these dependencies, we will denote $I(A; \hat{B}) = J(p_{A,B}, p_{\hat{B}|B})$. Consequently, given a prior $p_{A,B}$ linking the private attributes A and the data B , the privacy mapping $p_{\hat{B}|B}$ minimizing the inference threat subject to a distortion constraint is obtained as the solution to the following convex optimization problem [7]¹

$$\begin{aligned} & \underset{p_{\hat{B}|B} \in \text{Simplex}}{\text{minimize}} && J(p_{A,B}, p_{\hat{B}|B}) \\ & \text{s.t. } \mathbb{E}_{p_{B, \hat{B}}} [d(B, \hat{B})] \leq \Delta, \end{aligned} \quad (1)$$

where Simplex denotes the probability simplex ($\sum_x p(x) = 1$, $p(x) \geq 0 \forall x$).

Sparsity of the privacy mapping: When applying the aforementioned privacy-accuracy framework to large data, we encounter a challenge of scalability. Designing the privacy mapping requires characterizing the value of $p_{\hat{B}|B}(\hat{b}|b)$ for all possible pairs $(b, \hat{b}) \in \mathcal{B} \times \hat{\mathcal{B}}$, i.e. solving the convex optimization problem over $|\mathcal{B}||\hat{\mathcal{B}}|$ variables. When $\hat{\mathcal{B}} = \mathcal{B}$, and the size of the alphabet $|\mathcal{B}|$ is large, solving the optimization over $|\mathcal{B}|^2$ variables may become intractable.

¹Solving Optimization (1) for different values of Δ allows to generate the whole privacy-utility curve, as in Fig. 2. Any point on or above this curve is achievable. Using this curve, one can efficiently solve the related problem of maximizing utility given a constraint on privacy: for a given privacy requirement, the curve gives the operating point corresponding to the smallest loss in utility.

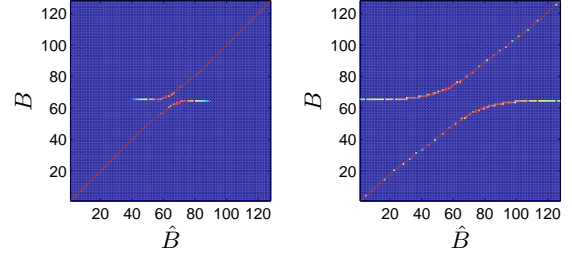


Figure 1: Optimal mappings for Toy example

A natural question is whether $\hat{\mathcal{B}} = \mathcal{B}$ is a necessary assumption to achieve the optimal privacy-utility trade-off. In other words, does Optimization (1) need to be solved over $|\mathcal{B}|^2$ variables to achieve optimality? The following theoretical toy example motivates a sparser approach to the design of the privacy mapping.

Example: Let $A \in \{0, 1\}$, and $B \in \{1, 2, \dots, 2^m\}$, and define the joint distribution $p_{A,B}$ such that $p(A = 0) = p(A = 1) = \frac{1}{2}$, and for $i \in \{1, 2, \dots, 2^m\}$, let $p(B = i|A = 0) = \frac{1}{2^{m-1}}$ if $i \leq 2^{m-1}$, 0 otherwise, and let $p(B = i|A = 1) = \frac{1}{2^{m-1}}$ if $i > 2^{m-1}$, 0 otherwise. For this example, the privacy threat is the worse it could be, as observing B determines deterministically the value of the private random variable A (equivalently, $I(A; B) = H(A) = 1$). In Fig. 2, we consider an l_2 distortion measure $d(B; \hat{B}) = (B - \hat{B})^2$ and illustrate the optimal mappings solving Problem ((1)) for different distortion values. For small distortions, the red diagonal in Figure 2 shows that most points $B = b$ are only mapped to themselves $\hat{B} = b$. As we increase the distortion level, each point $B = b$ gets mapped to a larger number of points $\hat{B} = \hat{b}$.

This theoretical example, as well as experiments on real world datasets such as the census data have shown that the optimal privacy preserving mapping may turn out to be sparse, in the sense that the support of $p_{\hat{B}|B}(\hat{b}|b)$ may be of much smaller size than \mathcal{B} , and may differ for different values of $B = b$. We propose to exploit sparsity properties of the privacy mapping to speed up the computation, by picking the support of $p_{\hat{B}|B}(\hat{b}|b)$, i.e. the set of points $\hat{B} = \hat{b}$ to which $B = b$ can be mapped with a non-zero probability, in an iterative greedy way. Although we a priori motivate the sparse approach using one theoretical example, and limited empirical evidence from some datasets, our experimental results demonstrate, a posteriori, that the sparse approach performs close to optimally on other datasets, and thus justifies empirically its relevance.

3 Sparse and Greedy Algorithm

Before we describe our algorithm, we rewrite Optimization (1) compactly. Let \mathbf{X} be a $n \times n$ matrix of optimized variables, whose entries are defined as

$x_{i,j} = p_{\hat{B}|B}(\hat{b}_i | b_j)$, and let \mathbf{x}_j be the j -th column of \mathbf{X} . To highlight the optimization aspect of our problem, we write the objective function $J(p_{A,B}, p_{\hat{B}|B})$ as a function $f(\mathbf{X})$, with the understanding that f depends on $p_{A,B}$, which is not optimized, and on \mathbf{X} , which is optimized. Similarly the distortion constraint can be written as $\sum_{j=1}^n \mathbf{d}_j^\top \mathbf{x}_j \leq \Delta$, where each $\mathbf{d}_j = p_B(b_j)(d(\hat{b}_1, b_j), d(\hat{b}_2, b_j), \dots, d(\hat{b}_n, b_j))^\top$ is a vector of length n that represents the distortion metric scaled by the probability of the corresponding symbol b_j . The marginal of B is computed as $p_B(b_j) = \sum_a p_{A,B}(a, b_j)$. Finally, the simplex constraint can be written as $\mathbf{1}_n^\top \mathbf{x}_j = 1$ for all j , where $\mathbf{1}_n$ is an all-ones vector of length n . Given the new notation, our original problem (1) can be written compactly as:

$$\begin{aligned} & \underset{\mathbf{X}}{\text{minimize}} && f(\mathbf{X}) \\ & \text{subject to} && \sum_{j=1}^n \mathbf{d}_j^\top \mathbf{x}_j \leq \Delta \\ & && \mathbf{1}_n^\top \mathbf{x}_j = 1 \quad \forall j = 1, \dots, n \\ & && \mathbf{X} \geq 0 \end{aligned} \quad (2)$$

where $\mathbf{X} \geq 0$ is an entry-wise inequality.

3.1 Franke-Wolfe Linearization

The optimization problem (1) has linear constraints but its objective function is non-linear. In this paper, we solve the problem as a sequence of linear programs, also known as the *Franke-Wolfe method*. Each iteration ℓ of the method consists of three major steps. First, we compute the gradient $\nabla_{\mathbf{X}} f(\mathbf{X}_{\ell-1})$ at the solution from the previous step $\mathbf{X}_{\ell-1}$. The gradient is a $n \times n$ matrix \mathbf{C} , where $c_{i,j} = \frac{\partial}{\partial x_{i,j}} f(\mathbf{X}_{\ell-1})$ is a partial derivative of the objective function with respect to the variable $x_{i,j}$. Second, we find a feasible solution \mathbf{X}' in the direction of the gradient. This problem is solved as a linear program with the same constraint as the original problem:

$$\begin{aligned} & \underset{\mathbf{X}}{\text{minimize}} && \sum_{j=1}^n \mathbf{c}_j^\top \mathbf{x}_j \\ & \text{subject to} && \sum_{j=1}^n \mathbf{d}_j^\top \mathbf{x}_j \leq \Delta \\ & && \mathbf{1}_n^\top \mathbf{x}_j = 1 \quad \forall j = 1, \dots, n \\ & && \mathbf{X} \geq 0 \end{aligned} \quad (3)$$

where \mathbf{c}_j is the j -th column of \mathbf{C} . Finally, we find the minimum of f between $\mathbf{X}_{\ell-1}$ and \mathbf{X}' , \mathbf{X}_ℓ , and make it the current solution. Since f is convex, this minimum can be found efficiently by ternary search. The minimum is also feasible because the feasible region is convex, and both \mathbf{X}' and $\mathbf{X}_{\ell-1}$ are feasible.

Algorithm 1 SPPM: Sparse privacy preserving maps

Input: Starting point \mathbf{X}_0 , number of steps L

for all $\ell = 1, 2, \dots, L$ **do**

$\mathbf{C} \leftarrow \nabla_{\mathbf{X}} f(\mathbf{X}_{\ell-1})$

$\mathcal{V} \leftarrow \text{DWD}$

 Find a feasible solution \mathbf{X}' in the direction of the gradient \mathbf{C} :

$$\begin{aligned} & \underset{\mathbf{X}}{\text{minimize}} && \sum_{j=1}^n \mathbf{c}_j^\top \mathbf{x}_j \\ & \text{subject to} && \sum_{j=1}^n \mathbf{d}_j^\top \mathbf{x}_j \leq \Delta \\ & && \mathbf{1}_n^\top \mathbf{x}_j = 1 \quad \forall j = 1, \dots, n \\ & && \mathbf{X} \geq 0 \\ & && x_{i,j} = 0 \quad \forall (i,j) \notin \mathcal{V} \end{aligned} \quad (4)$$

Find the minimum of f between $\mathbf{X}_{\ell-1}$ and \mathbf{X}' :

$$\gamma^* \leftarrow \arg \min_{\gamma \in [0,1]} f((1-\gamma)\mathbf{X}_{\ell-1} + \gamma\mathbf{X}') \quad (5)$$

$$\mathbf{X}_\ell \leftarrow (1-\gamma^*)\mathbf{X}_{\ell-1} + \gamma^*\mathbf{X}'$$

Output: Suboptimal feasible solution \mathbf{X}_L

3.2 Sparse Approximation

The linear program (3) has n^2 variables and therefore is hard to solve when n is large. In this section, we propose an incremental solution to this problem, which is defined only on a subset of *active variables* $\mathcal{V} \subseteq \{1, 2, \dots, n\} \times \{1, 2, \dots, n\}$. The active variables are the non-zero variables in the solution to the problem (3). Therefore, solving (3) on active variables \mathcal{V} is equivalent to restricting all inactive variables to zero. The corresponding linear program is shown in (4) in Algorithm 1. This linear program has only $|\mathcal{V}|$ variables. Now the challenge is in finding a good set of active variables \mathcal{V} . This set should be small, and such that the solutions of (3) and (4) are close.

We grow the set \mathcal{V} greedily using the dual linear program of (4). In particular, we incrementally solve the dual by adding most violated constraints, which corresponds to adding most beneficial variables in the primal. The dual of (4) is (6) in Algorithm 2, where $\lambda \in \mathbb{R}$ is a variable associated with the distortion constraint and $\mu \in \mathbb{R}^n$ is vector of n variables associated with the simplex constraints. Given a solution (λ^*, μ^*) to the dual, the most violated constraint for a given j is the one that minimizes:

$$c_{i,j} - \lambda^* d_{i,j} - \mu_j^*. \quad (8)$$

This quantity, called the *reduced cost*, has an intuitive interpretation. We choose an example i in the direc-

Algorithm 2 DWD: Dantzig-Wolfe decomposition

Initialize the set of active variables:

$$\mathcal{V} \leftarrow \{(1, 1), (2, 2), \dots, (n, n)\}$$

while the set \mathcal{V} grows **do**

Solve the master problem for λ^* and μ^* :

$$\begin{aligned} \underset{\lambda, \mu}{\text{maximize}} \quad & \lambda \Delta + \sum_{j=1}^n \mu_j \\ \text{subject to} \quad & \lambda \leq 0 \\ & \lambda d_{i,j} + \mu_j \leq c_{i,j} \quad \forall (i, j) \in \mathcal{V} \end{aligned} \quad (6)$$

for all $j = 1, 2, \dots, n$ **do**

Find the most violated constraint in the master problem for fixed j :

$$i^* = \arg \min_i [c_{i,j} - \lambda d_{i,j} - \mu_j] \quad (7)$$

if $(c_{i^*,j} - \lambda d_{i^*,j} - \mu_j < 0)$ **then**

$\mathcal{V} \leftarrow \mathcal{V} \cup \{(i^*, j)\}$

Output: Active variables \mathcal{V}

tion of the steepest gradient of $f(\mathbf{X})$, so $c_{i,j}$ is small; which is close to j , so $d_{i,j}$ is close to zero (as $\lambda^* \leq 0$). The search for the most violated constraint leverages the problem structure. Therefore, our approach can be viewed as an instance of *Dantzig-Wolfe decomposition*.

The pseudocode of our search procedure is in Algorithm 2. This is an iterative algorithm, where each iteration consists of three steps. First, we solve the reduced dual linear program (6) on active variables. Second, for each point j , we identify a point i^* that minimize the reduced cost. Finally, if the pair (i^*, j) corresponds to a violated constraint, we add it to the set of active variables \mathcal{V} .

The pseudocode of our final solution is in Algorithm 1. We refer to Algorithm 1 as *Sparse Privacy Preserving Mappings (SPPM)*, because of the mappings learned by the algorithm. Algorithm 2 is a subroutine of Algorithm 1, which identifies the set of active variables \mathcal{V} . SPPM is parameterized by the number of iterations L .

3.3 Convergence

Algorithm SPPM is a gradient descent method. In each iteration ℓ , we find a solution \mathbf{X}' in the direction of the gradient at the current solution $\mathbf{X}_{\ell-1}$. Then we find the minimum of f between $\mathbf{X}_{\ell-1}$ and \mathbf{X}' , and make it the next solution \mathbf{X}_ℓ . By assumption, the initial solution \mathbf{X}_0 is feasible in the original problem (1). The solution \mathbf{X}' to the LP (4) is always feasible in (1), because it satisfies all constraints in (1), and some additional constraints $x_{i,j} = 0$ on inactive variables. After the first iteration of SPPM, \mathbf{X}_1 is a convex combina-

tion of \mathbf{X}_0 and \mathbf{X}' . Since the feasible region is convex, and both \mathbf{X}_0 and \mathbf{X}' are feasible, \mathbf{X}_1 is also feasible. By induction, all solutions \mathbf{X}_ℓ are feasible.

The value of $f(X_\ell)$ is guaranteed to monotonically decrease with ℓ . When the method converges, $f(X_\ell) = f(X_{\ell-1})$. The convergence rate of the Frank-Wolfe algorithm is $O(1/L)$ in the worst case [5].

3.4 Computational Efficiency

The computation time of our method is dominated by the search for n^2 violated constraints in Algorithm 2. To search efficiently, we implement the following speedup in the computation of the gradients $c_{i,j}$. The marginal and conditional distributions:

$$\begin{aligned} p_{\hat{B}}(\hat{b}) &= \sum_{a,b} p_{A,B}(a,b) p_{\hat{B}|B}(\hat{b} | b) \\ p_{\hat{B}|A}(\hat{b} | a) &= \frac{\sum_b p_{A,B}(a,b) p_{\hat{B}|B}(\hat{b} | b)}{\sum_b p_{A,B}(a,b)} \end{aligned}$$

are precomputed, because these terms are common for all elements of \mathbf{C} . Then each gradient is computed as:

$$\begin{aligned} \frac{\partial}{\partial p(\hat{b}_i | b_j)} J(p_{a,b}, p_{\hat{b}|b}) &= \sum_a p(a, b_j) \log \frac{p(\hat{b}_i | a)}{p(\hat{b}_i)} \\ &+ \sum_a p(a, \hat{b}) \left(\frac{p(b_j | a)}{p(\hat{b}_i | a)} - \frac{p(a, b_j)}{p(\hat{b}_i)} \right). \end{aligned}$$

Since all marginals and conditionals are precomputed, each gradient can be computed in $O(|A|)$ time.

The space complexity of our method is $O(|\mathcal{V}|)$, because we operate only on active variables \mathcal{V} .

We point out that the complexity of the algorithm is closely linked to the sparsity of the optimal solution, which itself is related to the value of the distortion constraint Δ . This means that some distortion regimes may not be achievable with a given computational budget. Therefore one has to reduce Δ in order to have a sparser solution. In practice however, we did not run into problems, and were able to generate mappings efficiently even when high distortion was needed to drive the mutual information close to 0.

4 Evaluation

4.1 Datasets

Census Dataset: The *Census* dataset is a sample of the United States population from 1994, and contains both categorical and numerical features. Each entry in the dataset contains features such as age, workclass, education, gender, and native country, as well as income category (smaller or larger than 50k per year). For our purposes, we consider the information to be

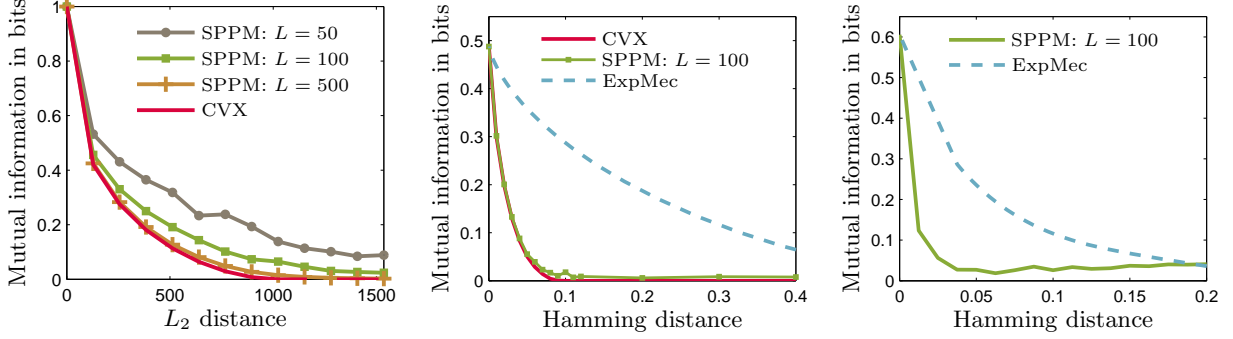


Figure 2: **Left.** Effect of parameters on privacy-distortion tradeoff. Synthetic data. **Middle.** Privacy distortion tradeoff. Census data. **Right.** Privacy distortion tradeoff. Movie data.

released publicly as the seven attributes shown in Table 1, while the income category is the private information to be protected. In this dataset, roughly 76% of the people have an income smaller than 50k. Due to the categorical nature of these features, a natural distortion metric for this data is the Hamming distance, and thus we use this metric in our experiments on this data. Fig. 3 shows that the user profile consisting of these 7 attributes can be a threat to income; the ROC curve illustrates the success rate of a simple classifier that tries to guess a user’s income category when there is no privacy protection.

Movie Dataset: The well known *MovieLens* dataset [1] consists of 1M ratings of 6K users on 4K movies. Each movie comes annotated with metadata indicating its genre. In MovieLens, there are 19 genres, that we expanded as follows. We gathered the more extended set of 300 genre tags from Netflix. From these, we select those that appear in at least 5% of movies, yielding 31 genres. For user j , we compute the preference for genre i as the probability that the user chooses a movie from the genre times the reciprocal of the number of movies in that genre. We capture the user profile using a binary vector of length 31; the bits corresponding to the six most preferred genres are set to one. We treat the preference vector as public but the gender of the user as private. The fact that this profile can be a threat to gender is illustrated in Fig. 4 which shows the success of a classifier that tries to guess gender when there is no privacy protection. Once more, as the features are categorical, we use the Hamming distance for our evaluations on this dataset.

Synthetic Dataset: We consider synthetic data as well since this allows us to freely vary the problem size. The input distribution is specified in our example in Sec. 2, namely the private attribute is a binary variable $A \in \{0, 1\}$, and the public attribute B is perfectly correlated with A . By varying the parameter m as defined in the example, we modify the size of the alphabet of B , which allow us to asses the scalability.

The distortion metric is the squared l_2 distance.

4.2 Benchmarks

Optimal mapping: The optimal mapping is the solution to (1); for small scale problems we were able to compute this using CVX without running out of memory. On our server, we could solve optimally (1) with alphabet size up to $|\mathcal{B}| = 2^{12} = 4096$.

Exponential Mechanism: The differential privacy metric is most commonly used in a database privacy setting, in which an analyst asks a query on a private database of size n containing data from n users. The privacy-preserving mechanism, which computes and releases the answer to the query, is designed to satisfy differential privacy under a given notion of neighboring databases. In the strong setting of *local* differential privacy [16], users do not trust the entity collecting the data in a database, thus each user holds his data locally, and passes it through a differentially private mechanism before releasing it to the untrusted entity. In this case, the privacy-preserving mechanism works on a database of size $n = 1$, and all possible databases are considered to be neighbors. This local differential privacy setting, based on input perturbation at the user end, is comparable to our local privacy setting, where user data is distorted before its release, but it differs from our setting by the privacy metric that the privacy mechanism is required to satisfy. More precisely, the local differential privacy setting considers a database of size 1 which contains the vector b of a user. The local differentially private mechanism p^{DP} satisfies $p^{DP}(\hat{b}|b) \leq e^\epsilon p^{DP}(\hat{b}|b'), \forall b, b' \in \mathcal{B}$ and $\forall \hat{b} \in \hat{\mathcal{B}}$.

As the non-private data in our 3 datasets is categorical, we focus on the *exponential mechanism* [19], a well-known mechanism that preserves differential privacy for non-numeric valued queries. More precisely, in our experiments, we use the exponential mechanism $p^{DP}(\hat{b}|b)$ that maps b to \hat{b} with a probability that decreases exponentially with the distance $d(\hat{b}, b)$

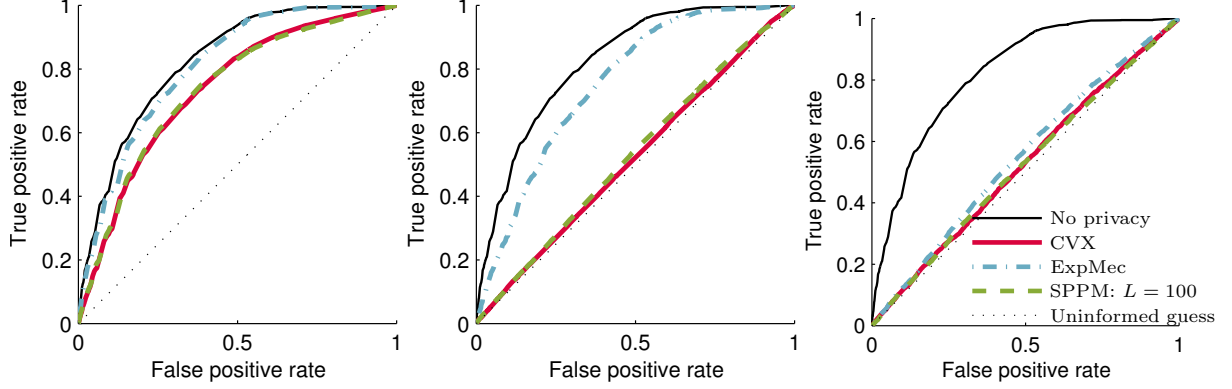


Figure 3: ROC for Naive Bayes Classifier with $\Delta = 0.02$ (Left), 0.14 (Middle) and 0.44 (Right). Census Data.

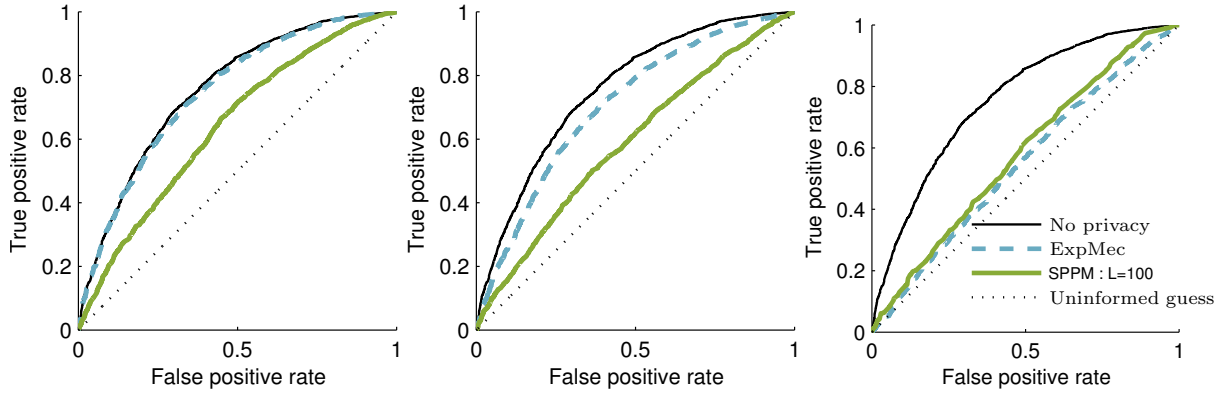


Figure 4: ROC for Logistic Regression with $\Delta = 0.04$ (Left), 0.13 (Middle) and 0.22 (Right). Movie Data.

$p^{DP}(\hat{b}|b) \propto \exp(-\beta d(\hat{b}, b))$, where $\beta \geq 0$. Let $d_{\max} = \sup_{b, \hat{b} \in \mathcal{B}} d(\hat{b}, b)$. This exponential mechanism satisfies $(2\beta d_{\max})$ -local differential privacy. Intuitively, the distance $d(\hat{b}, b)$ represents how appealing substituting \hat{b} for b is: the larger the distance $d(\hat{b}, b)$, the less appealing the substitution. To make a fair comparison between the exponential mechanism and SSPM, the distance $d(\hat{b}, b)$ used to define the exponential mechanism will be the same as the distance used in the distortion constraint of the optimization problem (1). In Section 4.3, $d(\hat{b}, b)$ will be the Hamming distance for experiments on the census and the movie datasets, and the squared l_2 distance for the synthetic datasets.

In [7], it was shown that in general, differential privacy with some neighboring database notion, does not guarantee low information leakage $I(A; \hat{B})$, for all priors $p_{A,B}$. However, it was also shown in [18], that *strong* ϵ -differential privacy, i.e. ϵ -differential privacy under the neighboring notion that all databases are neighbors, implies that $I(A; \hat{B}) \leq \epsilon$. Local differential privacy is a particular case of strong differential privacy. Consequently, the mutual information between private A and the distorted \hat{B}^{DP} resulting from the exponential mechanism p^{DP} will be upperbounded as

$I(A; \hat{B}^{DP}) \leq 2\beta d_{\max}$. We acknowledge that differential privacy was not defined with the goal of minimizing mutual information. However, regardless of the mechanism, mutual information is a relevant privacy metric [7, 18, 24, 25, 22, 12], thus we can compare these 2 algorithms with respect to this metric.

4.3 Results

Parameter Choices. Using synthetic data, we explore the privacy distortion tradeoff curve (in Fig. 2 **Left**) for different values of our algorithm's parameter L . First we observe that for small values of distortions, the difference between the various curves is insignificant, which suggests that in this regime the optimal mapping is indeed sparse. Second, as we increase L accuracy improves as we approach the optimal solution. Since the gain of using $L = 500$ compared to $L = 100$ seems small, and using 100 rather than 500 approximations is clearly much faster, we elect to use $L = 100$ for further experiments.

Privacy Performance. We provide two ways of comparing SSPM and our benchmarks on two datasets. We first compare them in Fig. 2 terms of tradeoff between privacy leakage, as measured by $I(A; \hat{B})$, and

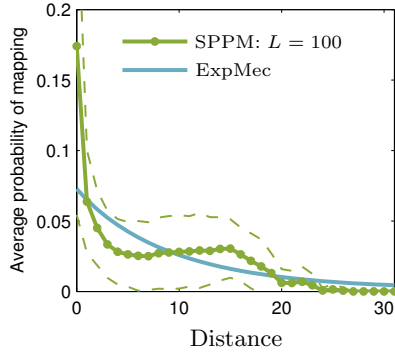


Figure 5: Behavior of SPPM and ExpMec. Synthetic data.

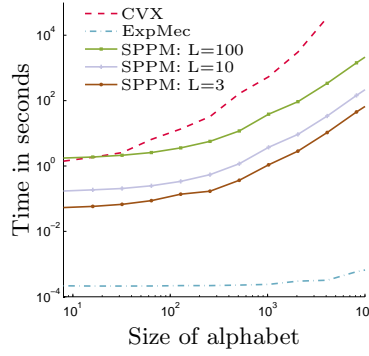


Figure 6: Time complexity and parameter sensitivity. Synthetic data.

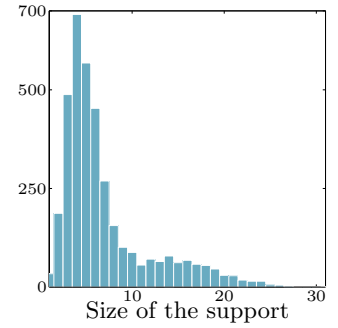


Figure 7: Histogram on size of support for mapping. Movie data, initial alphabet size 3717.

distortion. Then we draw another more general, meaningful, and fair comparison, in Fig. 3 and Fig. 4, by comparing the ability of SSPM and our benchmarks to defeat two different inference algorithms that would try to infer private data by using the privatized data $\hat{\mathcal{B}}$. ROC curves are agnostic and meaningful measures privacy that do not a priori favor a privacy metric (differential privacy or mutual information).

We start by illustrating the privacy versus distortion tradeoff for SPPM and our benchmarks, on the Census dataset, in which each user is represented by a vector of 7 attributes. We do not consider all possible values of this vector, as it would be prohibitive for an exact solver and prevent us from comparing to an optimal solution. Instead, we restrict the alphabet $|\mathcal{B}|$ to the 300 most probable vectors of 7 attributes, since we are mainly focused on a relative comparison. In Fig. 2(Middle) we see that SPPM is nearly indistinguishable from the optimal solution whereas the exponential mechanism (ExpMec) is much further away. To bring the mutual information down from 0.5 to 0.07, SPPM needs 0.05 distortion to achieve perfect privacy, while ExpMec needs more than 8 times as much. Note that for a given level of distortion, e.g. 0.1, 0.2, SPPM achieves much better privacy than ExpMec as the mutual information is significantly lower.

Next we consider the Movie dataset which is one order of magnitude larger than the Census dataset. The results in Fig. 2(Right) mirror what we observed with the Census data, namely that a given level of privacy can be achieved with less distortion using SPPM as opposed to ExpMec. For example, to reduce our privacy leakage metric from 0.6 to 0.05, SPPM requires roughly 0.03 distortion whereas ExpMec needs approximately 0.17, nearly 6 times as much. For both the Census and Movie datasets, SPPM can achieve perfect or near-perfect privacy with a small distortion.

Differential privacy does not aim to minimize mutual

information, which explains why ExpMec does not perform as well as SSPM in Fig. 2. Another metric to gage the success of our privacy mapping is to consider its impact on a classifier attempting to infer the private attribute. Recall that the goal of our mapping is to weaken any classifier that threatens to infer the private attribute. First, we consider a simple Naive Bayes classifier that analyzes the Census data to infer each user's income category. We quantify the classifier's success, in terms of true positives and false negatives (in an ROC curve) in Fig. 3. Recall that in an ROC curve, the $y = x$ line corresponds to a blind classifier that is no better than an uninformed guess. The weaker a classifier the closer it is to this line, and it becomes useless when it matches this line. We consider three bounds on distortion that allow us to explore the extremes of nearly no distortion ($\Delta = 0.02$ in Fig. 3(a)), a large amount of distortion ($\Delta = 0.44$ in Fig. 3(c)), and something in between ($\Delta = 0.14$ in Fig. 3(b)). In the case of small distortion, all algorithms make modest improvements over the no privacy case. However even in this scenario, SPPM performs close to optimal, unlike ExpMec that only slightly outperforms the no privacy case. With only a very small amount of distortion, not even the optimal solution can render the classifier completely useless (i.e. equivalent to the blind uninformed guess). On the other hand, when a large distortion is permitted, then all algorithms do naturally well (Fig. 3(c)). For a value of Δ between these extremes, SPPM is close to optimal, while ExpMec can only weaken the classifier a little.

In a second scenario, we study a logistic regression classifier that analyzes the movie dataset to infer gender. We focus on logistic regression for movie data because it has been shown to be an effective classifier for inferring gender [27]. Again we see in Fig. 4 that the findings essentially mimic those of the previous case. We thus conclude that, for a given distortion budget, SPPM is more successful against inference

threats because it can diminish the success of a classifier more than the exponential mechanism. In other words, SPPM can provide more privacy than the exponential mechanism for the same level of distortion.

Why is it that SPPM consistently outperforms ExpMec? Fig. 5 illustrates how these mappings work. We plot the average probability of being mapped to the k th closest point, together with the standard error that measures the variability among points. In ExpMec, the probability of mapping one point to another decreases exponentially with distance and the same mapping is applied to all points (in other words the standard deviation is null). With SPPM, many points are mapped to themselves. This means that the privacy of these points cannot be improved within the distortion constraint. This can happen if a given profile is already very private, or if there is no hope of providing privacy. In both cases, the distortion budget should not be wasted on such profiles, and SPPM is able to detect these cases and save its budget for other cases. ExpMec *wastes* some distortion on those points which are forced to be mapped to close neighbours. The key difference between SPPM and ExpMec is that SPPM is not required to apply the same mapping to each user, and can thus personalize the distortion, or adapt it as needed. This flexibility leads to a fair bit of variance as seen in Fig. 5. Because the probability of mapping to another point does not decrease with the distance from that point, we see a non-monotonically decreasing curve with distance: some points are better off being mapped to far neighbors rather than close ones; e.g. users whose profile is hard to disguise.

To further understand the effect of the mappings that SPPM proposes, we examine which features in a user profile are impacted most by our distortions. In Tab. 1 we list the mutual information between a single public feature (e.g., Education) and the private attribute we wish to hide (e.g., income category). We see that $I(A; F)$ is largest for Education, Marital status and Occupation, indicating these features are the most correlated with the private attribute. This is intuitive as, for example, more highly educated people tend to make larger salaries. The table shows that these 3 features experience the largest reduction in mutual information after distortion, indicating that SPPM spent its distortion budget on the biggest threats. This intuitive property shows that the mappings learned depend on the underlying prior distribution in a *smart* way, such that with limited distortion budget the priority is on the biggest privacy threats. Another point can be easily seen in the movie data in Tab. ??; here the single features mutual information are rather low, whereas the mutual information of the full vector of features is significant. This means that a big part of the privacy threat comes from co-occurrence of features rather than individual features. This explains

Feature F	Examples	$I(A; F)$	
		before	after
Age	10-20, 20-30,...	0.1064	0.0408
Education	Bachelor, PhD,...	0.1502	0.0702
Marital status	Divorced, Married,...	0.1241	0.0440
Occupation	Manager, Scientist,...	0.1126	0.0367
Race	Black, White,...	0.0192	0.0084
Gender	Female, Male	0.0123	0.0082
Country	Mexico, USA,...	0.0470	0.0203

Table 1: Mutual information between private attribute A and public attributes F before and after SPPM on Census dataset.

why it is not enough to simply reduce the mutual information of a single feature to obtain good privacy.

Scalability. Having established good privacy performance, we now assess how the runtime performance scales in terms of the size of the problem, shown in Fig. 6. We fix the distortion constraint to be proportional to the size of the problem, in order to keep a similar difficulty as we grow the size. As stated in Sec. 3.1, the time complexity is linear in L . This trend is evident as we observe the gaps between the lines for $L = 3, 10$, and 100 . Importantly, we see that our method scales with problem size better than the optimal solution. We observe that the computation speed of the exponential mechanism is very quick, and note that indeed this is one of the salient properties of this mechanism. Overall, this figure shows that our method is indeed tractable and can compute the distortion maps within a few minutes for problems whose alphabet size is on the order of tens of thousands.

Recall that we designed for computation efficiency by smartly selecting a limited number of alternate user profiles to map each original profile to. This corresponds to the support size of $p(\hat{B}|b_i)$ for all b_i . In Fig. 7 we show the histogram of the support sizes for our SPPM mapping on the movie data. Although the initial alphabet size is very large (above 3500), for most users we don’t consider more than 10 alternate profiles, and in the worst case we consider up to 30. This amounts to huge savings in computation and memory, without sacrificing privacy.

5 Conclusion

In this paper, for the first time, we apply large scale LP optimization techniques to the problem of data distortion for privacy. We show that our privacy-preserving mappings can be close to optimal, and consistently outperform a state of the art technique called the Exponential Mechanism. Our solution achieves better privacy with less distortion than existing solutions, when privacy leakage is measured by a mutual information metric. We demonstrated that our method can scale, even for systems with many users and a large underlying alphabet that describes their profiles.

References

- [1] MovieLens Dataset, GroupLens Research. <http://www.grouplens.org/datasets/movielens/>, 2010.
- [2] Rakesh Agrawal and Ramakrishnan Srikant. Privacy-preserving data mining. *ACM Sigmod Record*, 2000.
- [3] B. Fung and K. Wang and R. Chen and P. Yu. Privacy Preserving Data Publishing: A Survey of Recent Developments. *ACM Computing Surveys*, 2010.
- [4] Siddhartha Banerjee, Nidhi Hegde, and Laurent Massoulié. The price of privacy in untrusted recommendation engines. In *IEEE Allerton*, 2012.
- [5] Dimitri Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 1999.
- [6] C. Jernigan and B. Mistree. Gaydar: Facebook Friendships expose sexual orientation. In *First Monday*, October 2009.
- [7] Flavio Calmon and Nadia Fawaz. Privacy against statistical inference. In *IEEE Allerton*, 2012.
- [8] Kenneth L. Clarkson. Coresets, sparse greedy approximation, and the frank-wolfe algorithm. *ACM Trans. Algorithms*, September 2010.
- [9] John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. Local privacy and statistical minimax rates. In *IEEE FOCS*, 2013.
- [10] C. Dwork, F. Mcsherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, 2006.
- [11] Cynthia Dwork. Differential privacy. In *Automata, Languages and Programming*. Springer, 2006.
- [12] A. Evfimievski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *ACM PODS*, 2003.
- [13] Martin Jaggi. *Sparse Convex Optimization Methods for Machine Learning*. PhD thesis, 2011.
- [14] Martin Jaggi. Revisiting frank-wolfe: Projection-free sparse convex optimization. *submitted*, 2012.
- [15] Lee K. Jones. A simple lemma on greedy approximation in hilbert space and convergence rates for projection pursuit regression and neural network training. *The Annals of Statistics*, 1992.
- [16] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM Jour. on Computing*, 2011.
- [17] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramanian. l-diversity: Privacy beyond k-anonymity. *ACM TKDD*, 2007.
- [18] Ali Makhdoumi and Nadia Fawaz. Privacy-utility tradeoff under statistical uncertainty. In *IEEE Allerton*, 2013.
- [19] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *FOCS*, 2007.
- [20] Nina Mishra and Mark Sandler. Privacy via pseudo-random sketches. In *ACM PODS*, 2006.
- [21] J. Otterbacher. Inferring gender of movie reviewers: exploiting writing style, content and metadata. In *CIKM*, 2010.
- [22] D. Rebollo-Monedero, J. Forné, and J. Domingo-Ferrer. From t-closeness-like privacy to postrandomization via information theory. *IEEE Trans. on Knowledge and Data Engineering*, 2010.
- [23] I. S. Reed. Information Theory and Privacy in Data Banks. In *AFIPS '73*, pages 581–587, 1973.
- [24] Salman Salamatian, Amy Zhang, Flavio du Pin Calmon, Sandilya Bhamidipati, Nadia Fawaz, Branislav Kveton, Pedro Oliveira, and Nina Taft. How to hide the elephant- or the donkey- in the room: Practical privacy against statistical inference for large data. In *IEEE GlobalSIP*, 2013.
- [25] L. Sankar, S. R. Rajagopalan, and H. V. Poor. Utility-privacy tradeoff in databases: An information-theoretic approach. *IEEE Trans. Inf. Forensics Security*, 2013.
- [26] Latanya Sweeney. k-anonymity: A model for protecting privacy. *Internat. Jour. Uncertainty, Fuzziness and Knowledge-Based Systems*, 2002.
- [27] U. Weinsberg and S. Bhagat and S. Ioannidis and N. Taft. BlurMe: Inferring and Obfuscating User Gender Based on Ratings. In *ACM RecSys*, 2012.
- [28] Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Jour. American Statistical Association*, 1965.
- [29] H. Yamamoto. A source coding problem for sources with additional outputs to keep secret from the receiver of wiretappers. *IEEE Trans. Information Theory*, 29(6), 1983.
- [30] Tong Zhang. Sequential greedy approximation for certain convex optimization problems. *IEEE Trans. Inf. Theor.*, 49(3):682–691, 2006.
- [31] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *ICML*, 2003.

Correlated Compressive Sensing for Networked Data

Tianlin Shi Da Tang Liwen Xu Thomas Moscibroda

The Institute for Theoretical Computer Science (ITCS)
Institute for Interdisciplinary Information Sciences
Tsinghua University, Beijing

Abstract

We consider the problem of recovering sparse correlated data on networks. To improve accuracy and reduce costs, it is strongly desirable to take the potentially useful side-information of network structure into consideration. In this paper we present a novel correlated compressive sensing method called **CorrCS** for networked data. By naturally extending Bayesian compressive sensing, we extract correlations from network topology and encode them into a graphical model as prior. Then we derive posterior inference algorithms for the recovery of jointly sparse and correlated networked data. First, we design algorithms to recover the data based on pairwise correlations between neighboring nodes in the network. Next, we generalize this model through a diffusion process to capture higher-order correlations. Both real-valued and binary data are considered. Our models are extensively tested on several real datasets from social and sensor networks and are shown to outperform baseline compressive sensing models in terms of recovery performance.

1 INTRODUCTION

Networked data, from domains such as social network of friends, hyper-linked networks of webpages and distributed network of sensors, are becoming increasingly pervasive and important in modern signal processing and machine learning. Recent research has demonstrated compelling approaches to extract useful information from these networked data, including latent structure of social links (Kemp et al., 2004), community detection (Fortunato, 2010), etc. However, with the massive amount of data generated at an exploding rate, conventional ways of collecting networked data are being challenged, particularly when measurements are expensive and/or data are redundant.

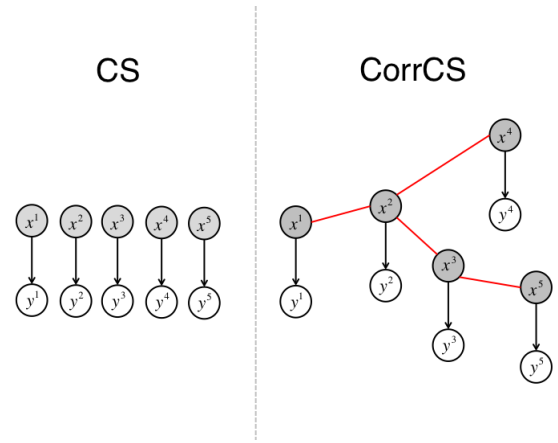


Figure 1: A graphical illustration of Compressive Sensing (CS) and Correlated Compressive Sensing (CorrCS) on networked data. In CS, each sparse signal x^i is recovered independently as y^i ; in CorrCS, they are recovered jointly with the network structure.

A significant finding of a large class of high-dimensional data over the last two decades is their inherent sparsity (Candès and Wakin, 2008). As a robust tool to leverage the sparsity, Compressive Sensing (CS) (Donoho, 2006) has been developed to collect high-dimensional sparse data from their low-dimensional projections. With sufficiently sparse signals, compressive sensing is guaranteed to recover the original signal from fewer samples required by Shannon-Nyquist limit (Candès, 2006). Compressive sensing has therefore been successfully applied to attack various problems of data collection in a wide range of fields, such as medical imaging (Lustig et al., 2008), low-level vision (Yang et al., 2008), etc.

Successful attempts have been made to apply compressive sensing to collect and analyze sparse networked data. For example, in studying large-scale sensor networks, the pioneering work by Luo et al. (2009) shows a successful scheme to efficiently gather spatially sparse sensor readings. For social networks, Compressive Network Anal-

ysis (Jiang et al., 2011) proposes a novel framework for clique detection. These approaches typically require finding a proper basis over the network topology so that data could be sparsely represented.

In many cases, however, it is unclear how the network topology could imply the sparse structure of the data, and therefore difficult to identify the sparse basis. Take the social network for example. It has been known that social influence and correlation exist at a large statistical level (Backstrom et al., 2006), but it turns out hard to directly model due to unobserved latent factors (Anagnostopoulos et al., 2008). An interesting question would be, under uncertainty about correlation of sparse data across the network, is it possible to seriously incorporate the network structure into compressive sensing and hopefully to improve recover performance?

In this paper, we present Correlated Compressive Sensing (CorrCS) to solve this problem. In particular, the setting in Figure 1 is considered. Each node i is equipped with a sensor, and we aim to recover the original high-dimensional data \mathbf{x}^i from its low-dimensional measurements \mathbf{y}^i . Instead of independently recover sparse signals, CorrCS incorporates side-information of the network structure and build correlation into signal modeling jointly with the inherent sparsity. By adopting a probabilistic approach, we show that it is possible to exploit the flexibility of graphical models to improve compressive sensing. Our approach is extensively tested on several real datasets, including product review data from social trust networks, social polling data and Air Quality Index from distributed sensors. The results show that CorrCS outperforms CS in terms of recover performance and demonstrates the usefulness of correlation in sensing networked data.

2 PRELIMINARY

In a typical sensing problem, the data of interest is regarded as a signal, which is a vector \mathbf{x} in a high-dimensional space \mathbb{R}^r . A measurement of \mathbf{x} is a low-dimensional vector $\mathbf{y} \in \mathbb{R}^m$ ($m \leq r$) from which the information of \mathbf{x} can be extracted. From a Bayesian point of view, this corresponds to inferring $p(\mathbf{x} | \mathbf{y}) \propto p(\mathbf{y} | \mathbf{x})p(\mathbf{x})$. The likelihood term $p(\mathbf{y} | \mathbf{x})$ describes a sensing model, which is the noisy measurement process, and the term $p(\mathbf{x})$ corresponds to a signal model, which represents the prior knowledge.

2.1 THE SENSING MODEL

A large body of sensing methods focus on the linear system

$$\mathbf{y} = V\mathbf{x} \quad (1)$$

with the goal to recover \mathbf{x} from \mathbf{y} accurately. However, since $r \gg m$, the inversion problem is highly ill-posed due to the undetermined solutions. One way to deal with the

uncertainty is to adopt a Gaussian generative model for \mathbf{y} as follows,

$$\mathbf{y} | \mathbf{x} \sim \mathcal{N}(V\mathbf{x}, \beta), \quad (2)$$

where β is the variance controlling the precision of measurement.

2.2 THE SIGNAL MODEL

Many natural signals $\mathbf{x} \in \mathbb{R}^r$ can be sparsely represented under some basis $\Phi = [\phi_1, \phi_2, \dots, \phi_K]$ as

$$\mathbf{x} = \Phi \mathbf{z} \quad (3)$$

where \mathbf{z} is the sparse coefficients such that $\|\mathbf{z}\|_0 = S \ll K$. Compressive sensing (Donoho, 2006; Candès and Wakin, 2008) shows that if the signal \mathbf{x} is sufficiently sparse, one can recover it effectively through minimizing the number of non-zero components in \mathbf{z} :

$$\begin{aligned} \min \quad & \|\mathbf{z}\|_0 \\ \text{s.t.} \quad & \mathbf{y} = M\mathbf{z} \end{aligned} \quad (4)$$

where $M = V\Phi$. In practice, it is often hard to solve the non-convex objective in (4) exactly, and a ℓ_1 relaxation is usually adopted, which corresponds to the basis pursuit (BP) algorithm (Chen et al., 1998). Candès et al. (2006) have proved that, under certain isometry properties, one can recover \mathbf{x} perfectly from $m = \Omega(S \log r)$ observations \mathbf{y} through BP.

To allow extra flexibility that we would exploit later, the framework of CS could be reformulated approximately as a Bayesian inference problem (Ji et al., 2008), and its goal is to design a signal model $p(\mathbf{z}; \Gamma)$ with some parameter Γ so that \mathbf{z} is controlled to be sufficiently sparse. Below we describe two common sparse signal models in social and sensor networks.

ℓ_1 prior for real-valued \mathbf{z} . Using sparsity-favoring Laplace priors on the coefficients (Babacan et al., 2010), one could use the following signal model:

$$p(\mathbf{z}; \lambda) = \frac{\lambda^{K/2}}{2^K} \exp\left(-\sqrt{\lambda}\|\mathbf{z}\|_1\right) \quad (5)$$

In practice, it is often inconvenient that the Laplace prior is not conjugate to the Gaussian signal model. However, one can show that (5) is equivalent to a hierarchical conjugate model parametrized by $\Gamma = (\{\gamma_k\}_{k=1}^K, \lambda)$ (Seeger and Nickisch, 2008).

$$\begin{aligned} p(\mathbf{z}; \Gamma) &= \prod_{k=1}^K \mathcal{N}(z_k; 0, \gamma_k) \\ p(\Gamma) &= \text{Gamma}(\gamma_k; 1, \frac{\lambda}{2}) \end{aligned} \quad (6)$$

The inference of \mathbf{z} for (6) can be efficient via the EM algorithm (Dempster et al., 1977).

Beta process for binary \mathbf{z} . An efficient way to characterize the sparsity of binary-valued coefficients is the Beta process (Paisley and Carin, 2009). In practice, a finite truncation of the process is used and leads to the following hierarchical conjugate model:

$$p(\mathbf{z}; \mathbf{\Gamma}) = \prod_{k=1}^K \text{Bernoulli}(z_k; \pi_k) \quad (7)$$

$$p(\mathbf{\Gamma}) = \prod_{k=1}^K \text{Beta}(\pi_k; \frac{a}{K}, b(1 - \frac{1}{K}))$$

where $\mathbf{\Gamma} = (\{\pi_k\}_{k=1}^K, a, b)$ and a, b are hyper-prior affecting sparsity. The exact posterior inference of (7) is intractable, but can be approximated through MCMC (Mohamed et al., 2011) or mean-field variational inference (Paisley and Carin, 2009).

3 CORRELATED COMPRESSIVE SENSING

Now, consider the problem of collecting data distributed on a network of n nodes. When the network structure is known, it can be described by a graph $G(V, E)$, where the edges have weight

$$E_{ij} = \begin{cases} w_{ij}, & \text{node } i \text{ and } j \text{ are adjacent} \\ 0, & \text{otherwise} \end{cases}, \quad (8)$$

where w_{ij} encodes the side-information about the correlation between the node i and j . In practice, such weight can either be collected directly from network or be computed through some metrics such as Pearson correlation and some function of geographic distance. When this weight is not exactly available, it is convenient to set $w_{ij} = 1$ for all edges uniformly. Let $(\mathbf{Z}, \mathbf{X}, \mathbf{Y}) = \{\mathbf{z}^i, \mathbf{x}^i, \mathbf{y}^i\}_{i=1}^n$, the Bayesian formulation of sensing is generalized as the following principle

$$p(\mathbf{Z}|\mathbf{Y}) \propto p(\mathbf{Z}) \prod_{i=1}^N \mathcal{N}(M^i \mathbf{z}^i, \beta). \quad (9)$$

Instead of applying compressive sensing independently to each node (i.e. $p(\mathbf{Z}) = \prod_i p(\mathbf{z}^i)$), Correlated Compressive Sensing (CorrCS) fuses the network structure G into recovery as side-information. To fulfill this goal, a joint distribution $p(\mathbf{Z})$ is explored in this section to capture the notion of joint sparsity and correlation.

3.1 PAIRWISE CORRELATION

The simplest form of correlation among networked data is pairwise according to the edge connecting neighboring nodes. Inspired by graphical models, we consider a range of pairwise Correlated Compressive Sensing

(CorrCS-Pair) that can be formulated as the Gibbs distribution

$$p(\mathbf{Z} | \mathbf{\Gamma}) \propto \exp \left(- \sum_i \mathcal{S}^i - c \sum_{(i,j) \in E} \mathcal{C}^{ij}(\mathbf{z}^i, \mathbf{z}^j) \right). \quad (10)$$

where \mathcal{S}^i is the sparsity of individual node \mathbf{z}^i controlled by hyper-parameter $\mathbf{\Gamma}$ and \mathcal{C}_{ij} models the pairwise correlation between two neighboring signal coefficients $\mathbf{z}_i, \mathbf{z}_j$. The parameter c controls prior on the strength of correlation.

Notice when $c = 0$, equation (10) reduces to independently applying BCS to each node. And the bigger c is, more correlation between neighboring nodes is favored over sparsity. On networked data with inherent correlation, we would expectedly improve the recovery performance with proper choice of positive c . However, if $c \rightarrow \infty$, the model would totally neglect sparsity, and therefore be undesirable. This variation of the recovery performance happens in actual experiments, as will be discussed later.

Below we consider two specific forms of CorrCS for real-valued and binary networked data. We discuss their inference algorithm in section 3.3.

Laplace-GRF Model. Assume $\mathbf{Z} = \mathbb{R}^{k \times n}$. Often we have the prior knowledge that neighboring sparse coefficient $\mathbf{z}^i, \mathbf{z}^j$ are close. This intuition leads to combining Laplace prior and Gaussian Random Field (GRF). Let

$$\mathcal{S}^i = \|\mathbf{z}^i\|_1$$

$$\mathcal{C}^{ij} = w_{ij} \|\mathbf{z}^i - \mathbf{z}^j\|^2.$$

The distribution $p(\mathbf{Z} ; \mathbf{\Gamma})$ is jointly Gaussian.

Beta-Ising Model. Assume $\mathbf{Z} = \{0, 1\}^{k \times n}$. This case is appealing for potential social network applications. For example, z_{ij} could be a latent feature indicating whether user i likes the product j . The Beta process (7) enforces the sparsity of binary coefficients. And based on similar idea of closeness, the Ising model shows a way to incorporate pairwise correlation into the model:

$$\mathcal{C}^{ij} = \sum_{k=1}^K w_{ij} (2z_{ik} - 1)(2z_{jk} - 1). \quad (11)$$

3.2 DIFFUSION PROCESS

We show that the pairwise correlation for real-valued signals can be generalized through a Diffusion Process (DP) on the graph G . The Correlated Compressive Sensing with Diffusion Process (CorrCS-DP) characterizes the covariant structure of the latent signals with a generative model, whose zeroth-order approximation is compressive sensing, and first-order approximation is pairwise CorrCS.

Diffusion Process. For any graph $G(V, E)$, a value function $f : V \rightarrow \mathbb{R}$ can be defined. Diffusion Process (DP)

is a natural class of stochastic processes on graphs that yields covariance structure of the function f (Kondor and Lafferty, 2002). First, we extend our value function as a function of time t : define $\mathbf{f}[t]$ be the snapshot vector of $[f(v_1), f(v_2), \dots, f(v_n)]$ at time t . Next, a diffusion generator H is defined as a Laplacian matrix of graph G as:

$$H_{ij} = \begin{cases} w_{ij} & \text{for } i \neq j \text{ and } j \in \mathcal{A}(i) \\ -\sum_{i'} w_{ii'} & \text{for } i=j \\ 0 & \text{otherwise.} \end{cases} \quad (12)$$

Then H is applied to the value function in the following way

$$\frac{\partial \mathbf{f}[t]}{\partial t} = \alpha H \mathbf{f}[t]. \quad (13)$$

Solving (13), we obtain

$$\mathbf{f}[t] = K \mathbf{f}[0], \quad (14)$$

where $K = \exp(\alpha t H)$ is the *Diffusion Kernel*. Notice that heat kernel is always invertible, which means given $\mathbf{f}[t]$ at any time t it is easy to compute $\mathbf{f}[0] = \exp(-\alpha t H) \mathbf{f}[t]$. Notice that when $t = 0$, $K = I$; when t is small, we have the first-order approximation $K = I - \alpha t H$.

Correlation via Diffusion. In CORRCS, we can define $f_k : V \rightarrow \mathbb{R}$ for each dimension of the features as $f_k^i = z_k^i$, so the snapshot \mathbf{f}_k is a vector. By studying the statistical characteristics of f_k , $k = 1, 2, \dots, K$, we can then build a correlated sparse signal model $p(\mathbf{Z})$, which is the core of CORRCS-DP.

Imagine that \mathbf{f}_k is generated through the following process: Initially $\mathbf{f}_k[0]$ is sparse. This means each entry $f_k^i[0]$ is distributed i.i.d as

$$f_k^i[0] \sim \mathcal{N}(0, \gamma_k^i), \quad \forall v_i \in V \quad (15)$$

Hyper-parameters γ_k^i control the sparsity of each entry. They are both drawn from hyper-priors according to equation (6).

A diffusion process is then run on the graph $G(V, E)$ and stops at some time t producing $\mathbf{f}_k[t] = K \mathbf{f}_k[0]$. Using the diffusion-based generative model, the inference problem (9) for real-valued signals becomes

$$p(\mathbf{Z} | \mathbf{Y}; \Gamma) \propto \Pr(\Gamma) \exp\left(-\frac{1}{2} \sum_k (\mathbf{f}^k)^\top (K^{-1})^\top D_k K^{-1} \mathbf{f}^k - \frac{1}{2} \beta \sum_i (\mathbf{y}_i - M_i \mathbf{x}_i)^\top (\mathbf{y}_i - M_i \mathbf{x}_i)\right) \quad (16)$$

where $D_k = \text{diag}(\gamma_k^i)^{-1}$. Notice that in this formulation, we no longer need constant c to control the extent of correlation, since it is directly induced by the prior uncertainty γ_k^i . A few observations can be made about the connection of CORRCS-DP to other compressive sensing methods summarized as follows.

Proposition 3.1. Using zeroth-order approximation $K := I$, where I is the identity matrix, CORRCS-DP subsumes BCS.

Proof. Straightforward. By replacing $K := I$ in (16), the posterior $p(\mathbf{X} | \mathbf{Y})$ is exactly the same as BCS. \square

Proposition 3.2. Using first-order approximation $K := I - \alpha t H$, CORRCS-DP reduces to Laplace-GRF model.

Proof. This claim is induced by the general property of Laplacian H that $\mathbf{f}_k^\top H \mathbf{f}^k = \sum_{ij} w_{ij} (f_k^i - f_k^j)^2$. Using the first order approximation, we have

$$-\log p(\mathbf{Z} | \mathbf{Y}; \Gamma) = \frac{1}{2} \sum_{ik} \left(\frac{\mathbf{f}_i^k}{\gamma_i^k}\right)^2 - \alpha t \sum_k (\mathbf{f}^k)^\top D_k H \mathbf{f}^k + \text{Const} \quad (17)$$

Let $d_i^k = 1/(\gamma_i^k)^2$. Notice H is a Laplacian matrix,

$$\begin{aligned} -(\mathbf{f}^k)^\top D_k^H \mathbf{f}^k &= \sum_{ij} w_{ij} \left(d_i^k (f_i^k)^2 + d_j^k (f_j^k)^2 - (d_i^k + d_j^k) f_i^k f_j^k \right) \\ &= \sum_{ij} w_{ij} \left(\frac{d_i + d_j}{2} \right) (f_i^k - f_j^k)^2 \\ &\quad - \frac{\alpha t}{2} \sum_i (H d^k)_i \end{aligned}$$

Let

$$\mathcal{S} = \frac{1}{2} \sum_{ik} (f_i^k)^2 \left((I - \frac{\alpha t}{2}) d^k \right)_i$$

and

$$\mathcal{C}_{ij} = w_{ij} \left(\frac{d_i + d_j}{2} \right) (f_i^k - f_j^k)^2,$$

then $p(\mathbf{X} | \mathbf{Y}; \Gamma) \propto \exp(-\mathcal{S} - \sum_{ij} \mathcal{C}_{ij})$ shows that CORR-DP reduces to a pairwise correlation model. \square

3.3 INFERENCE ALGORITHM

The exact inference of CORRCS is largely intractable due to two reasons. First, the signal model and the sensing model is not in same conjugate family. Second, even if $p(\mathbf{X}; \Gamma)$ is jointly Gaussian, in real applications either the number of nodes or the dimension of features is big.

Instead, we resort to approximation methods and develop the posterior inference based on Variational Bayes EM (Bernardo et al., 2003). In particular, we use the mean-field approximation $p(\mathbf{X} | \mathbf{Y}; \Gamma) = \prod_{ik} q(z_{ik}; \Gamma)$ and perform the following two-step scheme. In the E-step, we propagate information across nodes to spread correlation, which can be related to a message-passing process (Donoho et al., 2009); in the M-step, we update Γ to enforce sparsity. The details are outlined in Algorithm 1.

Algorithm 1 Correlated Compressive Sensing

```

1: Input: Network  $G(V, E)$ ,  $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n]$ , basis
    $\Phi$ , measurement matrices  $V^i$  and  $\text{iter}$ .
2: for  $i = 1 \rightarrow n$  do
3:   compute  $M^i = V^i \Phi$  for all  $i = 1, 2, \dots, n$ .
4:   initialize  $\mathbf{z}^i = (M^i)^\top \mathbf{y}_i$ .
5: end for
6: for  $j = 1 \rightarrow \text{iter}$  do
7:   for  $i = 1 \rightarrow n$  do
8:     for  $k = 1 \rightarrow k$  do
9:       Update factor  $q_k^i(\mathbf{z}_k^i)$  using equation (19), (20),
         (21).
10:    end for
11:  end for
12:  for  $i = 1 \rightarrow n$  do
13:    For binary case, use equation (22) to update  $\pi^i$ ;
    for real-valued case, use equation (24) to update
       $\gamma^i$ .
14:  end for
15: end for

```

E-Step: Spread Correlation.

In the pairwise case, the update algorithm in general is

$$q_i(\mathbf{z}_k^i) \propto \exp \left(\mathbb{E}_{q(\mathbf{z}_{-k}^i)} \left[\frac{1}{2} \beta \|\mathbf{y}^i - M_i \mathbf{z}_i\|^2 + \mathcal{S}_i \right] \right. \\ \left. + c \sum_{j \in \mathcal{A}(i)} \mathbb{E}_{q_j(\mathbf{z}^j)} [C_{ij} + C_{ji}] \right). \quad (18)$$

where \mathbf{z}_{-k}^i denotes all variables in \mathbf{z}^i except \mathbf{z}_k^i . Intuitively, the first expectation in (18) propagates information across dimensions of \mathbf{z}^i , while the second expectation in (18) spreads correlation among different nodes on the graph via the edges in between. Notice that for directed networks $w_{ij} \neq w_{ji}$, the information propagates forward and backward the edge in the same way, due to the symmetry of $C_{ij} + C_{ji}$.

Specifically, for Beta-Ising model,

$$q(z_k^i = 1) \propto \pi_k^i \exp \left(-\frac{1}{2} \beta (M_k^i)^\top (M_k^i) \right. \\ \left. - 2(M_k^i)^\top (\mathbf{y}^i - M_{-k}^i \mathbb{E}[\mathbf{z}_{-k}^i]) \right. \\ \left. + c \sum_{j \in \mathcal{A}(i)} (w_{ij} + w_{ji})(2z_k^i - 1) \right) \quad (19) \\ q(z_k^i = 0) \propto 1 - \pi_k^i.$$

Similarly for Laplace-GRF model, the mean-field update for each factor is $q_k^i(\mathbf{z}_k^i) = \mathcal{N}(\mu_k^i, \sigma_k^i)$, where

$$\sigma_k^i = (\beta (M_k^i)^\top M_k^i + 1/\gamma_k^i)^{-1} \\ \mu_k^i = \sigma_k^i \cdot \left(\beta (M_k^i)^\top (\mathbf{y}^i - M_{-k}^i \mu_{-k}^i) \right. \\ \left. + c \sum_{j \in \mathcal{A}(i)} (w_{ij} + w_{ji}) \mu_{-k}^j \right). \quad (20)$$

where μ_{-k}^i denotes all entries in μ^i except μ_k^i . For the extension of Laplace-GRF model, CorrCS-DP contains long-range interaction among the node, so all other nodes contribute to the distribution of the current node being updated. As in Laplace-GRF, we still have $q_k^i(\mathbf{z}_k^i) = \mathcal{N}(\mu_k^i, \sigma_k^i)$, but instead

$$\sigma_k^i = \left(\beta (M_k^i)^\top M_k^i + U_{ii}^k \right)^{-1} \\ \mu_k^i = \sigma_k^i \cdot \left(\beta (M_k^i)^\top (\mathbf{y}^i - M_{-k}^i \mu_{-k}^i) \right. \\ \left. + \sum_{j \in V} (U_{-i}^k + (U^k)_{-i}^\top)^\top \mathbf{u}_k^j \right). \quad (21)$$

where $U^k = K^{-T} D_k K^{-1}$ and $\mathbf{u}_k = [\mu_k^1, \mu_k^2, \dots, \mu_k^n]$.

Iteratively updating the factors according to equation (20), (19) and (21) guarantees convergence (Wainwright and Jordan, 2008).

M-Step: Update hyper-parameters. With the expectation of current belief about the signal to recover, we can further update the hyper-parameters Γ to enforce sparsity based on EM algorithm (Dempster et al., 1977). For Beta-Ising, we update the parameters of the Bernoulli prior π_k^i as follows

$$\pi_k^i \sim \frac{a/K + \mathbb{E}[z_k] - 1}{a/K + b(1 - 1/K) - 1}. \quad (22)$$

For Laplace-GRF, we update the global parameters

$$\gamma_k^i = -\frac{1}{2\lambda^2} + \sqrt{\frac{1}{4\lambda^2} + \frac{(\sigma_k^i + (\mu_k^i)^2)}{\lambda}}. \quad (23)$$

The update of γ_k^i in CorrCS-DP is similar to Laplace-GRF. Specifically, let Q^k be a diagonal matrix at time t such that $Q_{ii}^k = \sigma_{ii}^i$, compute $\tilde{Q}^k = K^{-1} Q^k K^{-\top}$, which can be regarded as the uncertainty about $\mathbf{f}_k[0]$. Then we modify (24) as

$$\gamma_k^i = -\frac{1}{2\lambda^2} + \sqrt{\frac{1}{4\lambda^2} + \frac{\tilde{Q}_{ii}^k}{\lambda}}. \quad (24)$$

Combining E-step and M-step, we can jointly optimize Γ and infer \mathbf{Z} , which eventually recovers the networked data on the graph G .

4 EXPERIMENT

We evaluate Correlated Compressive Sensing (CorrCS) empirically on real datasets from social and sensor networks with pairwise or Diffusion-like correlation.

4.1 SOCIAL NETWORK DATA WITH PAIRWISE CORRELATION

Using compressive sensing with pairwise correlation, we test the two recovery models Laplace-GRF and Beta-Ising on two datasets: product review on Epinion¹ and consumer polling data in Michigan.

4.1.1 Performance on different datasets

Michigan Polling Data. The social polling data is collected from a survey of consumers in Michigan with 500 monthly telephone calls from January, 1978 to December 2012. The data is real-valued aggregation of four hundred economic indices. It has been known that pairwise correlation exists in these indices. Using half of the dataset as past history, this pairwise correlation is computed through Pearson correlation and taken as the weight w_{ij} . Therefore, we establish a graph of features representing their inherent correlation. Furthermore, the data is continuous real values and to sparsify it, we use online sparse matrix factorization (Mairal et al., 2010) to find a set of sparse basis.

We test Laplace-GRF model on the Michigan polling data with $a = 1, b = 0.1$ and $c = 1$. As measurements, we randomly select a fraction of the polling data for each feature. We refer to the dimension of the selected data versus the dimension of the original data as measurement ratio. The performance is evaluated via Mean Squared Error (MSE) of the recover signal with respect to the original one. The MSE is normalized by the 2-norm of the original signal. For independent compressive sensing, we include two popular choices – Bayesian Compressive Sensing (Ji et al., 2008)(BCS) and Orthogonal Matching Pursuit (Tropp and Gilbert, 2007)(OMP) – as baseline algorithms. Notice that some baseline implementations of compressive sensing, such as Basis Pursuit (Chen et al., 1998)(BP), are too slow and impractical for networked data.

The result is reported in Figure 2. As we can see, Laplace-GRF outperforms BCS and OMP largely for small measurement ratios (less than 0.6). When the measurement ratio is large (exceed 0.6), Laplace-GRF will have a performance similar to BCS, and it still outperform OMP to a big extent. Correlation is less useful for large measurement ratios, mainly because the numbers of observations are already sufficient for sparse recovery. However, since in practice, people usually use a measure ratio in the rang $[0.2, 0.5]$, and rarely measure more than 60% of the data for recovering. Hence, this will not have strong influence on Laplace-GRF’s practical use.

To measure the superiority of Laplace-GRF for small measurement ratio, we could compare the minimum measurement ratio that is needed to achieve a fixed accuracy level (i.e. MSE level) for different models. Like in this

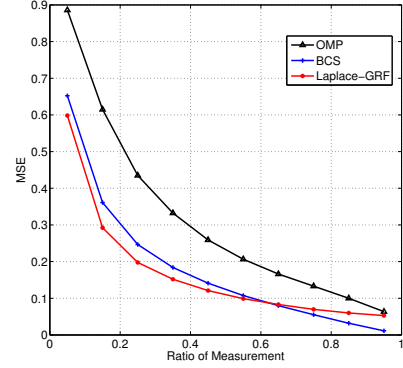


Figure 2: Performance of Laplace-GRF model versus the BCS model and OMP on the polling data.

dataset, we compare the minimum measurement ratio for different models to achieve a MSE not more than 0.2: our Laplace-GRF needs a measurement ratio of 0.23, BCS needs 0.32, while the most common baseline algorithm OMP needs 0.56. We could see that, to achieve this same recovery effect, Laplace-GRF needs 21% less measurements than BCS, and 59% less measurements than OMP. This result may imply that Laplace-GRF model could have valuable practical use since we could use a small number of measurements to recover a polling result, which is very close to the original ones. And by considering the inherent correlation between these indices, we could reduce the number of measurements by a ratio of 21% to recover a result with MSE 0.2 on this data, which means we could save a great deal of costs for this poll.

Epinion Data. The Epinion data is derived from the social product review network [Epinions](#) with 17,022 customers and 139,738 products. The graph G is built from the trust-list of all users: $w_{ij} = 1$ if and only if user i trusts user j , and therefore it is directed. To reduce the dimensionality of features, we select a subset of the most 100 popular products. Then z_{ij} represents whether customer i liked product j . The data Z is inherently sparse with only 5 to 10 nonzero per column, because the fraction of products rated by each customer is small. As measurements, each column z^i is projected to a low-dimensional space.

We test Beta-Ising model on the Epinion dataset against Bayesian Compressive Sensing (BCS) with beta prior. We choose $\lambda = 1, c = 0.3$. For the binary Epinion data, MSE is not a good choice because the data is zero almost everywhere. Instead, we regard the recovery as predicting label z_{ij} and use F1 score from classifier evaluations, which is the harmonic mean of precision (ratio of number of correct 1’s we recover over the total number of 1’s in our recovery result) and recall (ratio of number of correct 1’s we recover over the total number of 1’s in the ground truth).

¹Available at trustlet.org.

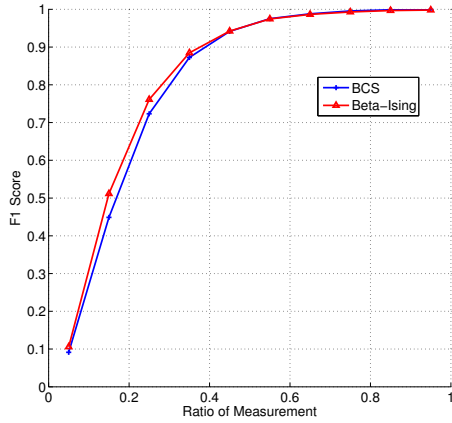


Figure 3: Performance of Beta-Ising model versus the Bayesian-Compressive Sensing on the Epinion data.

Figure 3 shows the result under F1 score. It can be seen that with correlation, the performance of BCS can be improved with a varying number of measurement ratios.

In our experiment, Beta-Ising model takes approximately the same amount of time for one iteration as BCS does, which indicates that it is a feasible and practical method for recovering binary networked data.

4.1.2 Sensitivity evaluation

Impact of parameters. In these two datasets, all parameters we could set are the parameter of the hierarchical conjugate prior a, b, λ and weight c . As has been discussed in BCS (Ji et al., 2008), our models are not sensitive to the parameters a, b, λ . So in this paper, we focus on the impact of the weight parameter c on the performance of the CorrCS models.

The choice of c is the key to CorrCS, which can be viewed as a regularization parameter controlling the tradeoff with sparsity. To evaluate the impact of the c on the performance, we test the variation of the behavior of the CorrCS model on different scenario. For example, consider the Beta-Ising model on the Epinion data, we compare the variation of the recovery F1 score corresponding to the change of c when some reasonable measurement ratios are selected (15%, 25% and 35%). Figure 4 shows that among all these 3 measurement ratios, the performance of this model will be improved when c starts to grow from 0, and will be demoted when c pass some specific values. This result accords with our discussion about c in Section 3.1. This kind of variation of the performance according to c 's variation is reasonable since different values of c imply different extents that we care about the inherent correlation between nodes in the network structure. This experiment show us that the performance will have the same trends of

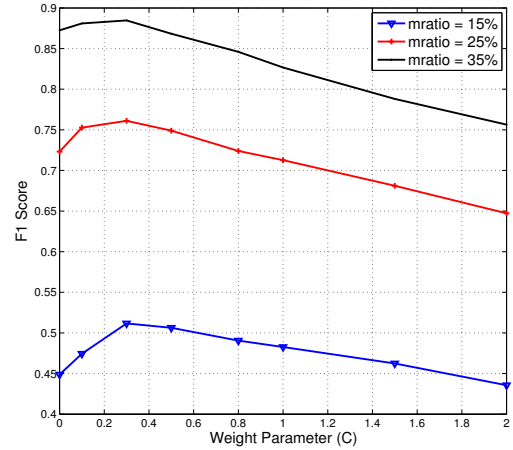


Figure 4: Relationship between the performance of the Beta-Ising model and the parameter c on different measure ratios.

variation on different measurement ratios when c is changing.

It is worth noting that among the 3 reasonable measurement ratios, the best c 's that will induce an optimal performance are very close. As shown in Figure 4, the model will possess an optimal performance when c is some value among 0.3. If $c = 0.3$, then the model will always have a nearly optimal recovery result as long as the measurement ratio is in a reasonable range. Therefore, in this model, we could choose an optimal choice of c that works well on all reasonable cases.

Impact of noise on the measured data. To test the robustness of these two models, we test the performance of our model on the two datasets when noises are added. More precisely, we add a Gaussian noise at each dimension of the observation y , where the standard deviation of this noise at each dimension is κ times the original value of this dimension, where κ is Signal-to-Noise Ratio. We test the performance of the two models as κ increases from 0 to 0.5 on different scenarios (i.e. different measure ratios and weight parameters). As can be seen in Figure 5, this two models possess strong robustness on the two datasets since even if κ goes to 0.5 the recovery result will not vary too much. The Beta-Ising model on the Epinion dataset has a slightly better robustness than that of the Laplace-GRF model on the polling dataset since it deals with binary variables, whose robust recovery turns out to be easier.

4.2 POLLUTION DATA FROM SENSOR NETWORK

The Beijing pollution data includes a network of 22 monitoring stations collecting data in the same time window

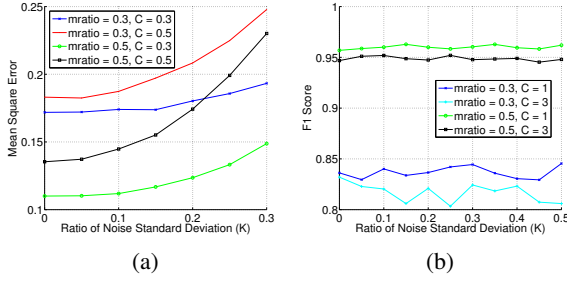


Figure 5: Robustness of `CorrCS` on the 2 datasets when some noise is added to the measure. **(a)**. Robustness of the Laplace-GRF model on the Polling dataset. **(b)**. Robustness of the Beta-Ising model on the Epinion dataset

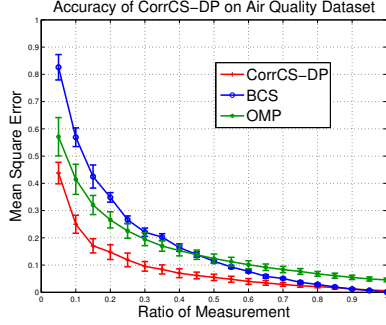


Figure 6: Performance of `CorrCS-DP` on Air Quality Dataset. Recovery accuracy of `CorrCS-DP` compared with BCS and OMP.

from Feb. 8th 2013 to Dec. 17 2013. The Air Quality Indexes (AQI) PM2.5 is recorded at an interval of 1 hour. The geography information of the sensors are available as GPS coordinates (g_i, l_i) , which we use to compute the edge weight of the sensor network through their euclidean distance $e_{ij} = \exp(-\theta\sqrt{(g_i - g_j)^2 + (l_i - l_j)^2})$.

The time sequence data is divided into 22 chunks, with 2 weeks of pollution data each chunk. The data is then split into two parts. Then we use 11 chunks to train a set of sparse basis using online sparse matrix factorization (Mairal et al., 2010), and also as cross-validation to find the best choice of diffusion time $t = 0.1$ and $c = 5$. The rest 11 chunks are used to test `CorrCS` and its counterparts. To simulate a real setting of measurement, we randomly select a portion of the samples as measurement and try to recover the rest. To test recovery accuracy in various situations, we change the ratio of measurement from 0 to 1 and compute the mean square error of the recovered signals.

Figure 6 and 7 shows the recovery accuracy and convergence rate of `CorrCS-DP` on the pollution dataset with comparison to the compressive sensing counterparts. The

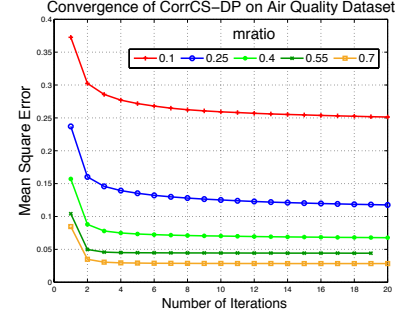


Figure 7: Convergence of recovery accuracy with number of iterations.

Objective MSE	<code>CorrCS-DP</code>	OMP	BCS
0.1	0.30	0.60	0.54
0.15	0.18	0.41	0.43
0.2	0.13	0.29	0.35

Table 1: Minimum measurement ratio to reach some particular values of MSE on the AQI dataset on different models

results are averaged over 10 independent runs. From Figure 6 we can see that `CorrCS` largely improves the recovery accuracy for various ratio of measurement, due to exploiting the correlation among different nodes. To measure the improvement of `CorrCS-DP` comparing to the other two models precisely, we could again compare the necessary minimum measurement ratio to reach some particular values of MSE on this data set on different models. As shown in Table 1, to reach the same MSE, `CorrCS-DP` could measure at least 40% less data than the other two models. Furthermore, as shown in Figure 7, `CorrCS` converges in about 3-4 steps. The convergence is faster than both OMP and BCS, and therefore demonstrates `CorrCS-DP` to be an efficacious and pragmatic correlated-recovery model.

5 RELATED WORK

Last two decades have witnessed significant advances in the theory and application of sensing sparse signals. Compressive sensing exploited the fact that natural signals are sparse and compressive under proper basis and designed sampling algorithms beyond the Nyquist-Shannon limit (Candès and Wakin, 2008). The theory of compressive sensing was developed by Candès et al. (2006) to explain this novel recovery performance. This theory was further improved by Candès and Tao (2006) to account for noisy measurements. The underlying property empowering sparse recovery is Restricted Isometry Property (RIP) of measurement matrices (Candès, 2008). On the algorithmic side, the first attempts to solve compressive sensing problems rely on ℓ_1 minimization under linear programming (Chen et al., 1998; Candès and Tao, 2005). Instead of optimizing with a large number of constraints, Orthog-

onal Matching Pursuit (Tropp and Gilbert, 2007) used a greedy heuristic to find solutions close to ℓ_0 optimum. To facilitate large-scale applications, Donoho et al. (2009) borrowed ideas from graphical models and derived a message passing algorithm for compressive sensing.

The Bayesian formulation of compressive sensing (BCS) is first proposed by Ji et al. (2008), which used a tractable conjugate Gamma prior on signal precision to enforce sparsity. It was shown that Bayesian compressive sensing allows uncertainty estimate and adaptive sampling. Based on BCS, Ji et al. (2009) developed a multi-tasking compressive sensing algorithm that allows simultaneously data collection from multiple sensors. Babacan et al. (2010) showed that stronger sparsity can be achieved for BCS with an conjugate prior on signal variance that is equivalent to the Laplace prior. As a counterpart of Laplace prior, beta prior is also commonly used (Paisley and Carin, 2009), with an additional latent variable controlling the support of signals. By comparing Laplace and Beta priors for sparse representation, Mohamed et al. (2011) concluded that Beta prior enforced stronger sparsity than the Laplace prior.

Real data is typically not sparse and therefore one must take effort in finding the appropriate basis. With a data-drive approach, dictionary learning for sparse basis originated from efforts in reproducing V1-like visual neurons through sparse coding (Olshausen and Field, 1997). Aharon et al. (2006) generalized the K-means clustering algorithm, and computed sparse decomposition by iteratively updating sparse coefficients and dictionary items. Mairal et al. (2009) proposed online dictionary learning methods, which leads to efficient computation of sparse coding.

Compressed sensing has find great applications in sensor networks. It was first successfully applied to network monitoring for optical and all-IP networks (Coates et al., 2007). In terms of data gathering, Luo et al. (2009) constructed a sensor network with a sink collecting compressed measurements, which is equivalent to a random matrix projection. Xu et al. (2013) considered more general compressed sparse functions for sparse representation of signals over graphs. Other than collecting data, compressive sensing was also used as a network analysis tool to identify social community on graphs (Jiang et al., 2011).

The topic of correlation in compressive sensing has been explored preliminarily in various ways. Shahrabi et al. (2011) used belief propagation to handle time-correlated signals with compressive sensing. Arildsen and Larsen (2014) explores the correlation of signal and measurement noise. In terms of networked data, Feizi et al. (2010) considers a distributed setting and a joint recovery model. As far as we know, our work is the first to explicitly incorporate graph structure as a hint of correlation.

Diffusion process has long been used as a general tool to capture correlation among data (Kondor and Lafferty,

2002). Ma et al. (2008) used diffusion process to model marketing candidate selection in social networks. The diffusion process may also be utilized to produce an representing wavelet basis on graphs and manifolds (Bremer et al., 2006). Using diffusion wavelets, it is possible to sparsify signals on different graph topologies and allow compressive sensing (Haupt et al., 2008). However, our correlated compressive sensing does not rely on the strong assumption that data on the network should be sparse under some basis, but rather weakly correlated.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we present Correlated Compressive Sensing (CorrCS) to leverage correlation among networked data and to empower better sparse recovery. Using a Bayesian approach, CorrCS allows flexible representation of prior knowledge about correlation via a graphical model. Two common types of correlation of networked data are considered: pairwise and diffusion-based. We have shown the diffusion-based formulation subsumes the pairwise case via a low-order approximation. Through extensive empirical evaluation on real data on social and sensor networks, we have demonstrated the advantage of correlated compressive sensing over its counterparts.

As future work, we are interested in showing bounds in its recovery performance to better understand its properties. Also we are interested in developing nonparametric extensions of the current approach to allow adaptive inference of key parameters and the basis for sparse representation.

7 ACKNOWLEDGEMENT

This work was supported in part by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, the National Natural Science Foundation of China Grant 61033001, 61361136003.

References

- Aharon, M., Elad, M., and Bruckstein, A. (2006). K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *Signal Processing, IEEE Transactions on*, 54(11):4311–4322.
- Anagnostopoulos, A., Kumar, R., and Mahdian, M. (2008). Influence and correlation in social networks. In *SIGKDD*, pages 7–15. ACM.
- Arildsen, T. and Larsen, T. (2014). Compressed sensing with linear correlation between signal and measurement noise. *Signal Processing*, 98:275–283.
- Babacan, S. D., Molina, R., and Katsaggelos, A. K. (2010). Bayesian compressive sensing using laplace priors. *Image Processing, IEEE Transactions on*, 19(1):53–63.
- Backstrom, L., Huttenlocher, D., Kleinberg, J., and Lan, X. (2006). Group formation in large social networks: membership, growth, and evolution. In *SIGKDD*, pages 44–54. ACM.

- Bernardo, J., Bayarri, M., Berger, J., Dawid, A., Heckerman, D., Smith, A., West, M., et al. (2003). The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures. *Bayesian statistics*, 7:453–464.
- Bremer, J. C., Coifman, R. R., Maggioni, M., and Szlam, A. D. (2006). Diffusion wavelet packets. *Applied and Computational Harmonic Analysis*, 21(1):95–112.
- Candès, E. J. (2006). Compressive sampling. In *Proceedings of the international congress of mathematicians*, volume 3, pages 1433–1452. Madrid, Spain.
- Candès, E. J. (2008). The restricted isometry property and its implications for compressed sensing. *Comptes Rendus Mathématique*, 346(9):589–592.
- Candès, E. J., Romberg, J., and Tao, T. (2006). Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *Information Theory, IEEE Transactions on*, 52(2):489–509.
- Candès, E. J. and Tao, T. (2005). Decoding by linear programming. *Information Theory, IEEE Transactions on*, 51(12):4203–4215.
- Candès, E. J. and Tao, T. (2006). Near-optimal signal recovery from random projections: Universal encoding strategies? *Information Theory, IEEE Transactions on*, 52(12):5406–5425.
- Candès, E. J. and Wakin, M. B. (2008). An introduction to compressive sampling. *Signal Processing Magazine, IEEE*, 25(2):21–30.
- Chen, S. S., Donoho, D. L., and Saunders, M. A. (1998). Atomic decomposition by basis pursuit. *SIAM journal on scientific computing*, 20(1):33–61.
- Coates, M., Pointurier, Y., and Rabbat, M. (2007). Compressed network monitoring for IP and all-optical networks. In *SIGCOMM*, pages 241–252. ACM.
- Dempster, A. P., Laird, N. M., Rubin, D. B., et al. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal statistical Society*, 39(1):1–38.
- Donoho, D. L. (2006). Compressed sensing. *Information Theory, IEEE Transactions on*, 52(4):1289–1306.
- Donoho, D. L., Maleki, A., and Montanari, A. (2009). Message-passing algorithms for compressed sensing. *PNAS*, 106(45):18914–18919.
- Feizi, S., Médard, M., and Effros, M. (2010). Compressive sensing over networks. In *Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on*, pages 1129–1136. IEEE.
- Fortunato, S. (2010). Community detection in graphs. *Physics Reports*.
- Haupt, J., Bajwa, W. U., Rabbat, M., and Nowak, R. (2008). Compressed sensing for networked data. *Signal Processing Magazine, IEEE*, 25(2):92–101.
- Ji, S., Dunson, D., and Carin, L. (2009). Multitask compressive sensing. *Signal Processing, IEEE Transactions on*, 57(1):92–106.
- Ji, S., Xue, Y., and Carin, L. (2008). Bayesian compressive sensing. *Signal Processing, IEEE Transactions on*, 56(6):2346–2356.
- Jiang, X., Yao, Y., Liu, H., and Guibas, L. (2011). Compressive network analysis. *arXiv preprint arXiv:1104.4605*.
- Kemp, C., Griffiths, T. L., and Tenenbaum, J. B. (2004). Discovering latent classes in relational data.
- Kondor, R. I. and Lafferty, J. (2002). Diffusion kernels on graphs and other discrete input spaces. In *ICML*, volume 2, pages 315–322.
- Luo, C., Wu, F., Sun, J., and Chen, C. W. (2009). Compressive data gathering for large-scale wireless sensor networks. In *MobiCom*, pages 145–156. ACM.
- Lustig, M., Donoho, D. L., Santos, J. M., and Pauly, J. M. (2008). Compressed sensing MRI. *Signal Processing Magazine, IEEE*, 25(2):72–82.
- Ma, H., Yang, H., Lyu, M. R., and King, I. (2008). Mining social networks using heat diffusion processes for marketing candidates selection. In *CIKM*, pages 233–242. ACM.
- Mairal, J., Bach, F., Ponce, J., and Sapiro, G. (2009). Online dictionary learning for sparse coding. In *ICML*, pages 689–696. ACM.
- Mairal, J., Bach, F., Ponce, J., and Sapiro, G. (2010). Online learning for matrix factorization and sparse coding. *JMLR*, 11:19–60.
- Mohamed, S., Heller, K., and Ghahramani, Z. (2011). Bayesian and L1 approaches to sparse unsupervised learning. *arXiv preprint arXiv:1106.1157*.
- Olshausen, B. A. and Field, D. J. (1997). Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision research*, 37(23):3311–3325.
- Paisley, J. and Carin, L. (2009). Nonparametric factor analysis with beta process priors. In *ICML*, pages 777–784. ACM.
- Seeger, M. W. and Nickisch, H. (2008). Compressed sensing and Bayesian experimental design. In *ICML*, pages 912–919. ACM.
- Shahrasbi, B., Talari, A., and Rahnavard, N. (2011). TC-CSBP: Compressive sensing for time-correlated data based on belief propagation. In *CISS*, pages 1–6. IEEE.
- Tropp, J. A. and Gilbert, A. C. (2007). Signal recovery from random measurements via orthogonal matching pursuit. *Information Theory, IEEE Transactions on*, 53(12):4655–4666.
- Wainwright, M. J. and Jordan, M. I. (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305.
- Xu, L., Qi, X., Wang, Y., and Moscibroda, T. (2013). Efficient data gathering using compressed sparse functions. In *INFOCOM, 2013 Proceedings IEEE*, pages 310–314. IEEE.
- Yang, J., Wright, J., Huang, T., and Ma, Y. (2008). Image super-resolution as sparse representation of raw image patches. In *CVPR*, pages 1–8. IEEE.

Improved Densification of One Permutation Hashing

Anshumali Shrivastava

Department of Computer Science
Computing and Information Science
Cornell University
Ithaca, NY 14853, USA
anshu@cs.cornell.edu

Ping Li

Department of Statistics and Biostatistics
Department of Computer Science
Rutgers University
Piscataway, NJ 08854, USA
pingli@stat.rutgers.edu

Abstract

The existing work on densification of one permutation hashing [24] reduces the query processing cost of the (K, L) -parameterized Locality Sensitive Hashing (LSH) algorithm with minwise hashing, from $O(dKL)$ to merely $O(d + KL)$, where d is the number of nonzeros of the data vector, K is the number of hashes in each hash table, and L is the number of hash tables. While that is a substantial improvement, our analysis reveals that the existing densification scheme in [24] is sub-optimal. In particular, there is not enough randomness in that procedure, which affects its accuracy on very sparse datasets.

In this paper, we provide a new densification procedure which is provably better than the existing scheme [24]. This improvement is more significant for very sparse datasets which are common over the web. The improved technique has the same cost of $O(d + KL)$ for query processing, thereby making it strictly preferable over the existing procedure. Experimental evaluations on public datasets, in the task of hashing based near neighbor search, support our theoretical findings.

1 Introduction

Binary representations are common for high dimensional sparse data over the web [8, 25, 26, 1], especially for text data represented by high-order n -grams [4, 12]. Binary vectors can also be equivalently viewed as sets, over the universe of all the features, containing only locations of the non-zero entries. Given two sets $S_1, S_2 \subseteq \Omega = \{1, 2, \dots, D\}$, a popular measure of similarity between sets (or binary vectors) is the *resemblance* R , defined as

$$R = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|} = \frac{a}{f_1 + f_2 - a}, \quad (1)$$

where $f_1 = |S_1|$, $f_2 = |S_2|$, and $a = |S_1 \cap S_2|$.

It is well-known that minwise hashing belongs to the *Locality Sensitive Hashing (LSH)* family [5, 9]. The method

applies a random permutation $\pi : \Omega \rightarrow \Omega$, on the given set S , and stores the minimum value after the permutation mapping. Formally,

$$h_\pi(S) = \min(\pi(S)). \quad (2)$$

Given sets S_1 and S_2 , it can be shown by elementary probability arguments that

$$Pr(h_\pi(S_1) = h_\pi(S_2)) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|} = R. \quad (3)$$

The probability of collision (equality of hash values), under minwise hashing, is equal to the similarity of interest R . This property, also known as the *LSH property* [14, 9], makes minwise hash functions h_π suitable for creating hash buckets, which leads to sublinear algorithms for similarity search. Because of this same LSH property, minwise hashing is a popular indexing technique for a variety of large-scale data processing applications, which include duplicate detection [4, 13], all-pair similarity [3], fast linear learning [19], temporal correlation [10], 3-way similarity & retrieval [17, 23], graph algorithms [6, 11, 21], and more.

Querying with a standard (K, L) -parameterized LSH algorithm [14], for fast similarity search, requires computing $K \times L$ min-hash values per query, where K is the number of hashes in each hash table and L is the number of hash tables. In theory, the value of KL grows with the data size [14]. In practice, typically, this number ranges from a few hundreds to a few thousands. Thus, processing a single query, for near-neighbor search, requires evaluating hundreds or thousands of independent permutations π (or cheaper universal approximations to permutations [7, 22, 20]) over the given data vector. If d denotes the number of non-zeros in the query vector, then the query preprocessing cost is $O(dKL)$ which is also the bottleneck step in the LSH algorithm [14]. Query time (latency) is crucial in many user-facing applications, such as search.

Linear learning with b -bit minwise hashing [19], requires multiple evaluations (say k) of h_π for a given data vector. Computing k different min-hashes of the test data costs $O(dk)$, while after processing, classifying this data vector

(with SVM or logistic regression) only requires a single inner product with the weight vector which is $O(k)$. Again, the bottleneck step during testing prediction is the evaluation of k min-hashes. Testing time directly translates into the latency of on-line classification systems.

The idea of storing k contiguous minimum values after one single permutation [4, 15, 16] leads to hash values which do not satisfy the LSH property because the hashes are not properly aligned. The estimators are also not linear, and therefore they do not lead to feature representation for linear learning with resemblance. This is a serious limitation.

Recently it was shown that a “rotation” technique [24] for densifying sparse sketches from one permutation hashing [18] solves the problem of costly processing with min-wise hashing (See Sec. 2). The scheme only requires a single permutation and generates k different hash values, satisfying the LSH property (i.e., Eq.(3)), in linear time $O(d + k)$, thereby reducing a factor d in the processing cost compared to the original minwise hashing.

Our Contributions: In this paper, we argue that the existing densification scheme [24] is not the optimal way of densifying the sparse sketches of one permutation hashing at the given processing cost. In particular, we provide a provably better densification scheme for generating k hashes with the same processing cost of $O(d + k)$. Our contributions can be summarized as follows.

- Our detailed variance analysis of the hashes obtained from the existing densification scheme [24] reveals that there is not enough randomness in that procedure which leads to high variance in very sparse datasets.
- We provide a new densification scheme for one permutation hashing with provably smaller variance than the scheme in [24]. The improvement becomes more significant for very sparse datasets which are common in practice. The improved scheme retains the computational complexity of $O(d + k)$ for computing k different hash evaluations of a given vector.
- We provide experimental evidences on publicly available datasets, which demonstrate the superiority of the improved densification procedure over the existing scheme, in the task of resemblance estimation and as well as the task of near neighbor retrieval with LSH.

2 Background

2.1 One Permutation Hashing

As illustrated in Figure 1, instead of conducting k independent permutations, *one permutation hashing* [18] uses only one permutation and partitions the (permuted) feature space into k bins. In other words, a single permutation π is used to first shuffle the given binary vector, and then the shuffled vector is binned into k evenly spaced bins. The

k minimums, computed for each bin separately, are the k different hash values. Obviously, empty bins are possible.

	Bin 0	Bin 1	Bin 2	Bin 3	Bin 4	Bin 5
$\pi(\Omega)$	0 1 2 3	4 5 6 7	8 9 10 11	12 13 14 15	16 17 18 19	20 21 22 23
	0 1 2 3	0 1 2 3	0 1 2 3	0 1 2 3	0 1 2 3	0 1 2 3
$\pi(S_1)$	0 0 0 0	0 <u>1</u> 0 1	0 0 0 0	0 0 <u>1</u> 1	<u>1</u> 0 1 0	0 <u>1</u> 1 0
$\pi(S_2)$	0 0 0 0	0 <u>1</u> 1 1	0 0 0 0	<u>1</u> 0 1 0	<u>1</u> 1 0 0	0 0 0 0
$OPH(S_1)$	E	1	E	2	0	1
$OPH(S_2)$	E	1	E	0	0	E

Figure 1: One permutation hashes [18] for vectors S_1 and S_2 using a single permutation π . For bins not containing any non-zeros, we use special symbol “E”.

For example, in Figure 1, $\pi(S_1)$ and $\pi(S_2)$ denote the state of the binary vectors S_1 and S_2 after applying permutation π . These shuffled vectors are then divided into 6 bins of length 4 each. We start the numbering from 0. We look into each bin and store the corresponding minimum non-zero index. For bins not containing any non-zeros, we use a special symbol “E” to denote empty bins. We also denote

$$M_j(\pi(S)) = \left\{ \pi(S) \cap \left[\frac{Dj}{k}, \frac{D(j+1)}{k} \right) \right\} \quad (4)$$

We assume for the rest of the paper that D is divisible by k , otherwise we can always pad extra dummy features. We define OPH_j (“OPH” for one permutation hashing) as

$$OPH_j(\pi(S)) = \begin{cases} E, & \text{if } \pi(S) \cap \left[\frac{Dj}{k}, \frac{D(j+1)}{k} \right) = \emptyset \\ M_j(\pi(S)) \bmod \frac{D}{k}, & \text{otherwise} \end{cases} \quad (5)$$

i.e., $OPH_j(\pi(S))$ denotes the minimum value in Bin j , under permutation mapping π , as shown in the example in Figure 1. If this intersection is null, i.e., $\pi(S) \cap \left[\frac{Dj}{k}, \frac{D(j+1)}{k} \right) = \emptyset$, then $OPH_j(\pi(S)) = E$.

Consider the events of “simultaneously empty bin” $I_{emp}^j = 1$ and “simultaneously non-empty bin” $I_{emp}^j = 0$, between given vectors S_1 and S_2 , defined as:

$$I_{emp}^j = \begin{cases} 1, & \text{if } OPH_j(\pi(S_1)) = OPH_j(\pi(S_2)) = E \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

Simultaneously empty bins are only defined with respect to two sets S_1 and S_2 . In Figure 1, $I_{emp}^0 = 1$ and $I_{emp}^2 = 1$, while $I_{emp}^1 = I_{emp}^3 = I_{emp}^4 = I_{emp}^5 = 0$. Bin 5 is only empty for S_2 and not for S_1 , so $I_{emp}^5 = 0$.

Given a bin number j , if it is not simultaneously empty

($I_{emp}^j = 0$) for both the vectors S_1 and S_2 , [18] showed

$$\Pr \left(OPH_j(\pi(S_1)) = OPH_j(\pi(S_2)) \mid I_{emp}^j = 0 \right) = R \quad (7)$$

On the other hand, when $I_{emp}^j = 1$, no such guarantee exists. When $I_{emp}^j = 1$ collision does not have enough information about the similarity R . Since the event $I_{emp}^j = 1$ can only be determined given the two vectors S_1 and S_2 and the materialization of π , one permutation hashing cannot be directly used for indexing, especially when the data are very sparse. In particular, $OPH_j(\pi(S))$ does not lead to a valid LSH hash function because of the coupled event $I_{emp}^j = 1$ in (7). The simple strategy of ignoring empty bins leads to biased estimators of resemblance and shows poor performance [24]. Because of this same reason, one permutation hashing cannot be directly used to extract random features for linear learning with resemblance kernel.

2.2 Densifying One Permutation Hashing for Indexing and Linear Learning

[24] proposed a “rotation” scheme that assigns new values to all the empty bins, generated from one permutation hashing, in an unbiased fashion. The rotation scheme for filling the empty bins from Figure 1 is shown in Figure 2. The idea is that for every empty bin, the scheme borrows the value of the closest non-empty bin in the clockwise direction (circular right hand side) added with offset C .

	Bin 0	Bin 1	Bin 2	Bin 3	Bin 4	Bin 5
$H(S_1)$	1+C	1	2+C	2	0	1
$H(S_2)$	1+C	1	0+C	0	0	1+2C

Figure 2: Densification by “rotation” for filling empty bins generated from one permutation hashing [24]. Every empty bin is assigned the value of the closest non-empty bin, towards right (circular), with an offset C . For the configuration shown in Figure 1, the above figure shows the new assigned values (in red) of empty bins after densification.

Given the configuration in Figure 1, for Bin 2 corresponding to S_1 , we borrow the value 2 from Bin 3 along with an additional offset of C . Interesting is the case of Bin 5 for S_2 , the circular right is Bin 0 which was empty. Bin 0 borrows from Bin 1 acquiring value $1 + C$, Bin 5 borrows this value with another offset C . The value of Bin 5 finally becomes $1 + 2C$. The value of $C = \frac{D}{k} + 1$ enforces proper alignment and ensures no unexpected collisions. Without this offset C , Bin 5, which was not simultaneously empty, after reassignment, will have value 1 for both S_1 and S_2 . This would be an error as initially there was no collision (note $I_{emp}^5 = 0$). Multiplication by the distance of the non-empty bin, from where the value was borrowed, ensures

that the new values of simultaneous empty bins ($I_{emp}^j = 1$), at any location j for S_1 and S_2 , never match if their new values come from different bin numbers.

Formally the hashing scheme with “rotation”, denoted by \mathcal{H} , is defined as:

$$\mathcal{H}_j(S) = \begin{cases} OPH_j(\pi(S)) & \text{if } OPH_j(\pi(S)) \neq E \\ OPH_{(j+t) \bmod k}(\pi(S)) + tC & \text{otherwise} \end{cases} \quad (8)$$

$$t = \min z, \quad \text{s.t.} \quad OPH_{(j+z) \bmod k}(\pi(S)) \neq E \quad (9)$$

Here $C = \frac{D}{k} + 1$ is a constant.

This densification scheme ensures that whenever $I_{emp}^j = 0$, i.e., Bin j is simultaneously empty for any two S_1 and S_2 under considerations, the newly assigned value mimics the collision probability of the nearest simultaneously non-empty bin towards right (circular) hand side making the final collision probability equal to R , irrespective of whether $I_{emp}^j = 0$ or $I_{emp}^j = 1$. [24] proved this fact as a theorem.

Theorem 1 [24]

$$\Pr(\mathcal{H}_j(S_1) = \mathcal{H}_j(S_2)) = R \quad (10)$$

Theorem 1 implies that \mathcal{H} satisfies the LSH property and hence it is suitable for indexing based sublinear similarity search. Generating KL different hash values of \mathcal{H} only requires $O(d + KL)$, which saves a factor of d in the query processing cost compared to the cost of $O(dKL)$ with traditional minwise hashing. For fast linear learning [19] with k different hash values the new scheme only needs $O(d+k)$ testing (or prediction) time compared to standard b -bit minwise hashing which requires $O(dk)$ time for testing.

3 Variance Analysis of Existing Scheme

We first provide the variance analysis of the existing scheme [24]. Theorem 1 leads to an unbiased estimator of R between S_1 and S_2 defined as:

$$\hat{R} = \frac{1}{k} \sum_{j=0}^{k-1} \mathbf{1}\{\mathcal{H}_j(S_1) = \mathcal{H}_j(S_2)\}. \quad (11)$$

Denote the number of simultaneously empty bins by

$$N_{emp} = \sum_{j=0}^{k-1} \mathbf{1}\{I_{emp}^j = 1\}, \quad (12)$$

where $\mathbf{1}$ is the indicator function. We partition the event $(\mathcal{H}_j(S_1) = \mathcal{H}_j(S_2))$ into two cases depending on I_{emp}^j . Let M_j^N (Non-empty Match at j) and M_j^E (Empty Match at j) be the events defined as:

$$M_j^N = \mathbf{1}\{I_{emp}^j = 0 \text{ and } \mathcal{H}_j(S_1) = \mathcal{H}_j(S_2)\} \quad (13)$$

$$M_j^E = \mathbf{1}\{I_{emp}^j = 1 \text{ and } \mathcal{H}_j(S_1) = \mathcal{H}_j(S_2)\} \quad (14)$$

Note that, $M_j^N = 1 \implies M_j^E = 0$ and $M_j^E = 1 \implies M_j^N = 0$. This combined with Theorem 1 implies,

$$\begin{aligned}\mathbb{E}(M_j^N | I_{emp}^j = 0) &= \mathbb{E}(M_j^E | I_{emp}^j = 1) \\ &= \mathbb{E}(M_j^E + M_j^N) = R \quad \forall j\end{aligned}\quad (15)$$

It is not difficult to show that,

$$\mathbb{E}(M_j^N M_i^N | i \neq j, I_{emp}^j = 0 \text{ and } I_{emp}^i = 0) = R\tilde{R},$$

where $\tilde{R} = \frac{a-1}{f+1-f2-a-1}$. Using these new events, we have

$$\hat{R} = \frac{1}{k} \sum_{j=0}^{k-1} [M_j^E + M_j^N] \quad (16)$$

We are interested in computing

$$\text{Var}(\hat{R}) = \mathbb{E} \left(\left(\frac{1}{k} \sum_{j=0}^{k-1} [M_j^E + M_j^N] \right)^2 \right) - R^2 \quad (17)$$

For notational convenience we will use m to denote the event $k - N_{emp} = m$, i.e., the expression $\mathbb{E}(\cdot | m)$ means $\mathbb{E}(\cdot | k - N_{emp} = m)$. To simplify the analysis, we will first compute the conditional expectation

$$f(m) = \mathbb{E} \left(\left(\frac{1}{k} \sum_{j=0}^{k-1} [M_j^E + M_j^N] \right)^2 \middle| m \right) \quad (18)$$

By expansion and linearity of expectation, we obtain

$$\begin{aligned}k^2 f(m) &= \mathbb{E} \left[\sum_{i \neq j} M_i^N M_j^N \middle| m \right] + \mathbb{E} \left[\sum_{i \neq j} M_i^E M_j^E \middle| m \right] \\ &+ \mathbb{E} \left[\sum_{i \neq j} M_i^E M_j^N \middle| m \right] + \mathbb{E} \left[\sum_{i=1}^k [(M_j^N)^2 + (M_j^E)^2] \middle| m \right]\end{aligned}$$

$M_j^N = (M_j^N)^2$ and $M_j^E = (M_j^E)^2$ as they are indicator functions and can only take values 0 and 1. Hence,

$$\mathbb{E} \left[\sum_{j=0}^{k-1} [(M_j^N)^2 + (M_j^E)^2] \middle| m \right] = kR \quad (19)$$

The values of the remaining three terms are given by the following 3 Lemmas; See the proofs in the Appendix.

Lemma 1

$$\mathbb{E} \left[\sum_{i \neq j} M_i^N M_j^N \middle| m \right] = m(m-1)R\tilde{R} \quad (20)$$

Lemma 2

$$\mathbb{E} \left[\sum_{i \neq j} M_i^E M_j^E \middle| m \right] = 2m(k-m) \left[\frac{R}{m} + \frac{(m-1)R\tilde{R}}{m} \right] \quad (21)$$

Lemma 3

$$\begin{aligned}\mathbb{E} \left[\sum_{i \neq j} M_i^E M_j^E \middle| m \right] &= (k-m)(k-m-1) \\ &\times \left[\frac{2R}{m+1} + \frac{(m-1)R\tilde{R}}{m+1} \right]\end{aligned}\quad (22)$$

Combining the expressions from the above 3 Lemmas and Eq.(19), we can compute $f(m)$. Taking a further expectation over values of m to remove the conditional dependency, the variance of \hat{R} can be shown in the next Theorem.

Theorem 2

$$\begin{aligned}\text{Var}(\hat{R}) &= \frac{R}{k} + A \frac{R}{k} + B \frac{R\tilde{R}}{k} - R^2 \\ A &= 2\mathbb{E} \left[\frac{N_{emp}}{k - N_{emp} + 1} \right] \\ B &= (k+1)\mathbb{E} \left[\frac{k - N_{emp} - 1}{k - N_{emp} + 1} \right]\end{aligned}\quad (23)$$

The theoretical values of A and B can be computed using the probability of the event $\text{Pr}(N_{emp} = i)$, denoted by P_i , which is given by Theorem 3 in [18].

$$P_i = \sum_{s=0}^{k-i} \frac{(-1)^s k!}{i!s!(k-i-s)!} \prod_{t=0}^{f_1+f_2-a-1} \frac{D(1 - \frac{i+s}{k}) - t}{D-t}$$

4 Intuition for the Improved Scheme

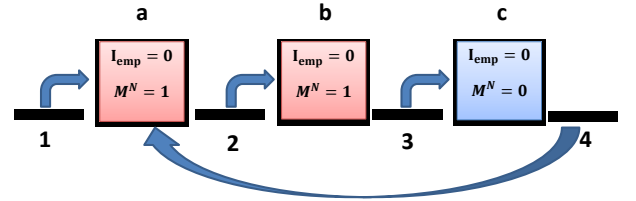


Figure 3: Illustration of the existing densification scheme [24]. The 3 boxes indicate 3 simultaneously non-empty bins. Any simultaneously empty bin has 4 possible positions shown by blank spaces. Arrow indicates the choice of simultaneous non-empty bins picked by simultaneously empty bins occurring in the corresponding positions. A simultaneously empty bin occurring in position 3 uses the information from Bin c. The randomness is in the position number of these bins which depends on π .

Consider a situation in Figure 3, where there are 3 simultaneously non-empty bins ($I_{emp} = 0$) for given S_1 and S_2 . The actual position numbers of these simultaneously non-empty bins are random. The simultaneously empty bins ($I_{emp} = 1$) can occur in any order in the 4 blank spaces. The arrows in the figure show the simultaneously

non-empty bins which are being picked by the simultaneously empty bins ($I_{emp} = 1$) located in the shown blank spaces. The randomness in the system is in the ordering of simultaneously empty and simultaneously non-empty bins.

Given a simultaneously non-empty Bin t ($I_{emp}^t = 0$), the probability that it is picked by a given simultaneously empty Bin i ($I_{emp}^i = 1$) is exactly $\frac{1}{m}$. This is because the permutation π is perfectly random and given m , any ordering of m simultaneously non-empty bins and $k - m$ simultaneously empty bins are equally likely. Hence, we obtain the term $\left[\frac{R}{m} + \frac{(m-1)R\tilde{R}}{m} \right]$ in Lemma 2.

On the other hand, under the given scheme, the probability that two simultaneously empty bins, i and j , (i.e., $I_{emp}^i = 1$, $I_{emp}^j = 1$), both pick the same simultaneous non-empty Bin t ($I_{emp}^t = 0$) is given by (see proof of Lemma 3)

$$p = \frac{2}{m+1} \quad (24)$$

The value of p is high because there is not enough randomness in the selection procedure. Since $R \leq 1$ and $R \leq R\tilde{R}$, if we can reduce this probability p then we reduce the value of $[pR + (1-p)R\tilde{R}]$. This directly reduces the value of $(k-m)(k-m-1) \left[\frac{2R}{m+1} + \frac{(m-1)R\tilde{R}}{m+1} \right]$ as given by Lemma 3. The reduction scales with N_{emp} .

For every simultaneously empty bin, the current scheme uses the information of the closest non-empty bin in the right. Because of the symmetry in the arguments, changing the direction to left instead of right also leads to a valid densification scheme with exactly same variance. This is where we can infuse randomness without violating the alignment necessary for unbiased densification. We show that randomly switching between left and right provably improves (reduces) the variance by making the sampling procedure of simultaneously non-empty bins more random.

5 The Improved Densification Scheme

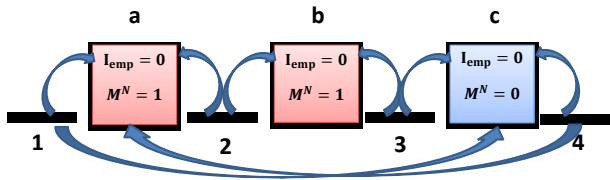


Figure 4: Illustration of the improved densification scheme. For every simultaneously empty bin, in the blank position, instead of always choosing the simultaneously non-empty bin from right, the new scheme randomly chooses to go either left or right. A simultaneously empty bin occurring at position 2 uniformly chooses among Bin a or Bin b.

Our proposal is explained in Figure 4. Instead of using the value of the closest non-empty bin from the right (circular),

we will choose to go either left or right with probability $\frac{1}{2}$. This adds more randomness in the selection procedure.

In the new scheme, we only need to store 1 random bit for each bin, which decides the direction (circular left or circular right) to proceed for finding the closest non-empty bin. The new assignment of the empty bins from Figure 1 is shown in Figure 5. Every bin number i has an i.i.d. Bernoulli random variable q_i (1 bit) associated with it. If Bin i is empty, we check the value of q_i . If $q_i = 1$, we move circular right to find the closest non-empty bin and use its value. In case when $q = 0$, we move circular left.

	Bin 0	Bin 1	Bin 2	Bin 3	Bin 4	Bin 5
Direction Bits (q)	0	1	0	0	1	1
$H^+(S_1)$	1+C	1	1+C	2	0	1
$H^+(S_2)$	0+2C	1	1+C	0	0	1+2C

Figure 5: Assigned values (in red) of empty bins from Figure 1 using the improved densification procedure. Every empty Bin i uses the value of the closest non-empty bin, towards circular left or circular right depending on the random direction bit q_i , with offset C .

For S_1 , we have $q_0 = 0$ for empty Bin 0, we therefore move circular left and borrow value from Bin 5 with offset C making the final value $1 + C$. Similarly for empty Bin 2 we have $q_2 = 0$ and we use the value of Bin 1 (circular left) added with C . For S_2 and Bin 0, we have $q_0 = 0$ and the next circular left bin is Bin 5 which is empty so we continue and borrow value from Bin 4, which is 0, with offset $2C$. It is a factor of 2 because we traveled 2 bins to locate the first non-empty bin. For Bin 2, again $q_2 = 0$ and the closest circular left non-empty bin is Bin 1, at distance 1, so the new value of Bin 2 for S_2 is $1 + C$. For Bin 5, $q_5 = 1$, so we go circular right and find non-empty Bin 1 at distance 2. The new hash value of Bin 5 is therefore $1 + 2C$. Note that the non-empty bins remain unchanged.

Formally, let $q_j = \{0, 1, 2, \dots, k-1\}$ be k i.i.d. Bernoulli random variables such that $q_j = 1$ with probability $\frac{1}{2}$. The improved hash function \mathcal{H}^+ is given by

$$\mathcal{H}_j^+(S) = \begin{cases} OPH_{(j-t_1) \bmod k}(\pi(S)) + t_1 C & \text{if } q_j = 0 \text{ and } OPH_j(\pi(S)) = E \\ OPH_{(j+t_2) \bmod k}(\pi(S)) + t_2 C & \text{if } q_j = 1 \text{ and } OPH_j(\pi(S)) = E \\ OPH_j(\pi(S)) & \text{otherwise} \end{cases} \quad (25)$$

where

$$t_1 = \min z, \quad s.t. \quad OPH_{(j-z) \bmod k}(\pi(S)) \neq E \quad (26)$$

$$t_2 = \min z, \quad s.t. \quad OPH_{(j+z) \bmod k}(\pi(S)) \neq E \quad (27)$$

with same $C = \frac{D}{k} + 1$. Computing k hash evaluations with \mathcal{H}^+ requires evaluating $\pi(S)$ followed by two passes over the k bins from different directions. The total complexity of computing k hash evaluations is again $O(d + k)$ which is the same as that of the existing densification scheme. We need an additional storage of the k bits (roughly hundreds or thousands in practice) which is practically negligible.

It is not difficult to show that \mathcal{H}^+ satisfies the LSH property for resemblance, which we state as a theorem.

Theorem 3

$$\Pr(\mathcal{H}_j^+(S_1) = \mathcal{H}_j^+(S_2)) = R \quad (28)$$

\mathcal{H}^+ leads to an unbiased estimator of resemblance \hat{R}^+

$$\hat{R}^+ = \frac{1}{k} \sum_{j=0}^{k-1} \mathbf{1}\{\mathcal{H}_j^+(S_1) = \mathcal{H}_j^+(S_2)\}. \quad (29)$$

6 Variance Analysis of Improved Scheme

When $m = 1$ (an event with prob $(\frac{1}{k})^{f_1+f_2-a} \simeq 0$), i.e., only one simultaneously non-empty bin, both the schemes are exactly same. For simplicity of expressions, we will assume that the number of simultaneous non-empty bins is strictly greater than 1, i.e., $m > 1$. The general case has an extra term for $m = 1$, which makes the expression unnecessarily complicated without changing the final conclusion.

Following the notation as in Sec. 3, we denote

$$M_j^{N+} = \mathbf{1}\{I_{emp}^j = 0 \text{ and } \mathcal{H}_j^+(S_1) = \mathcal{H}_j^+(S_2)\} \quad (30)$$

$$M_j^{E+} = \mathbf{1}\{I_{emp}^j = 1 \text{ and } \mathcal{H}_j^+(S_1) = \mathcal{H}_j^+(S_2)\} \quad (31)$$

The two expectations $\mathbb{E}\left[\sum_{i \neq j} M_i^{N+} M_j^{N+} \middle| m\right]$ and $\mathbb{E}\left[\sum_{i \neq j} M_i^{N+} M_j^{E+} \middle| m\right]$ are the same as given by Lemma 1 and Lemma 2 respectively, as all the arguments used to prove them still hold for the new scheme. The only change is in the term $\mathbb{E}\left[\sum_{i \neq j} M_i^{E+} M_j^{E+} \middle| m\right]$.

Lemma 4

$$\begin{aligned} \mathbb{E}\left[\sum_{i \neq j} M_i^{E+} M_j^{E+} \middle| m\right] &= (k-m)(k-m-1) \\ &\times \left[\frac{3R}{2(m+1)} + \frac{(2m-1)R\tilde{R}}{2(m+1)} \right] \end{aligned} \quad (32)$$

The theoretical variance of the new estimator \hat{R}^+ is given by the following Theorem 4.

Theorem 4

$$\begin{aligned} Var(\hat{R}^+) &= \frac{R}{k} + A^+ \frac{R}{k^2} + B^+ \frac{R\tilde{R}}{k^2} - R^2 \\ A^+ &= \mathbb{E}\left[\frac{N_{emp}(4k - N_{emp} + 1)}{2(k - N_{emp} + 1)}\right] \\ B^+ &= \mathbb{E}\left[\frac{2k^3 + N_{emp}^2 - N_{emp}(2k^2 + 2k + 1) - 2k}{2(k - N_{emp} + 1)}\right] \end{aligned} \quad (33)$$

The new scheme reduces the value of p (see Eq.(24)) from $\frac{2}{m+1}$ to $\frac{1.5}{m+1}$. As argued in Sec. 4, this reduces the overall variance. Here, we state it as theorem that $Var(\hat{R}^+) \leq Var(\hat{R})$ always.

Theorem 5

$$Var(\hat{R}^+) \leq Var(\hat{R}) \quad (34)$$

More precisely,

$$\begin{aligned} &Var(\hat{R}) - Var(\hat{R}^+) \\ &= \mathbb{E}\left[\frac{(N_{emp})(N_{emp} - 1)}{2k^2(k - N_{emp} + 1)}[R - R\tilde{R}]\right] \end{aligned} \quad (35)$$

The probability of simultaneously empty bins increases with increasing sparsity in dataset and the total number of bins k . We can see from Theorem 5 that with more simultaneously empty bins, i.e., higher N_{emp} , the gain with the improved scheme \mathcal{H}^+ is higher compared to \mathcal{H} . Hence, \mathcal{H}^+ should be significantly better than the existing scheme for very sparse datasets or in scenarios when we need a large number of hash values.

7 Evaluations

Our first experiment concerns the validation of the theoretical variances of the two densification schemes. The second experiment focuses on comparing the two schemes in the context of near neighbor search with LSH.

7.1 Comparisons of Mean Square Errors

We empirically verify the theoretical variances of \mathcal{R} and \mathcal{R}^+ and their effects in many practical scenarios. To achieve this, we extracted 12 pairs of words (which cover a wide spectrum of sparsity and similarity) from the web-crawl dataset which consists of word representation from 2^{16} documents. Every word is represented as a binary vector (or set) of $D = 2^{16}$ dimension, with a feature value of 1 indicating the presence of that word in the corresponding document. See Table 1 for detailed information of the data.

For all 12 pairs of words, we estimate the resemblance using the two estimators \mathcal{R} and \mathcal{R}^+ . We plot the empirical

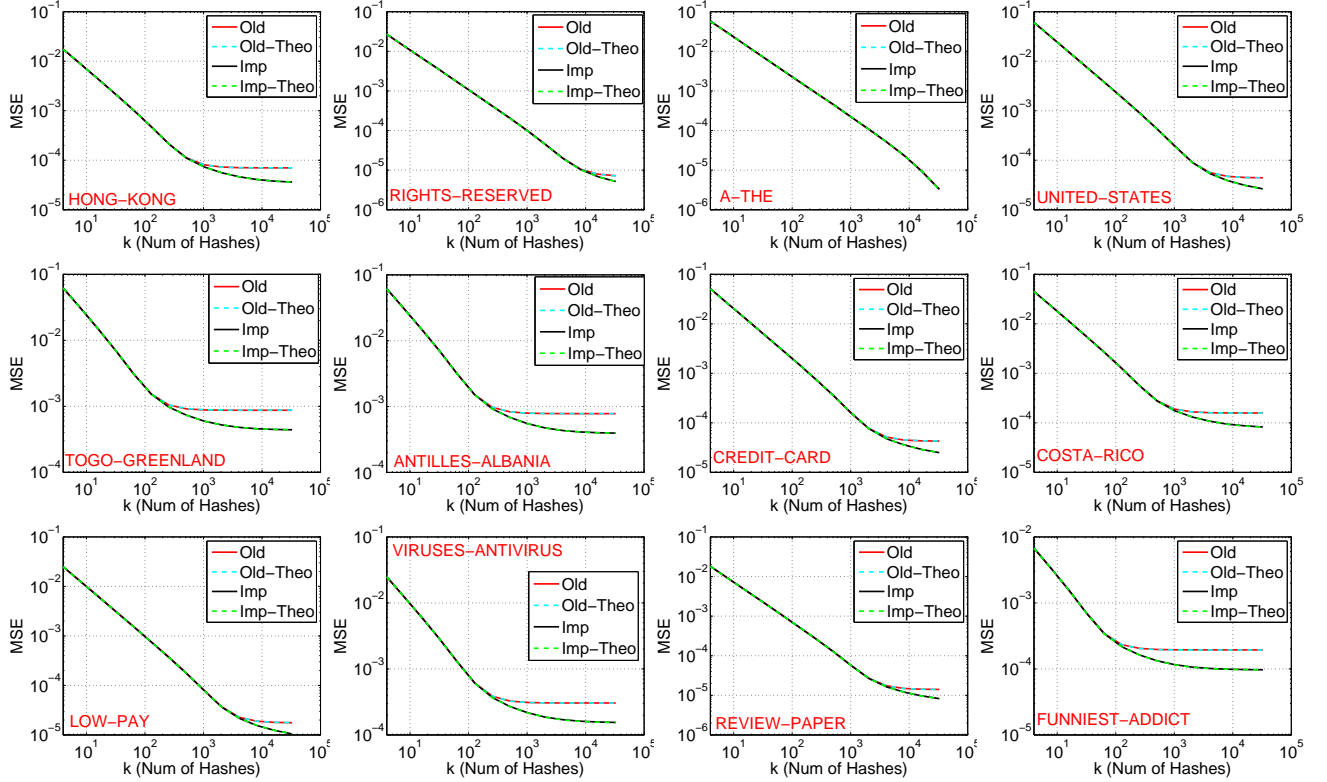


Figure 6: Mean Square Error (MSE) of the old scheme \hat{R} and the improved scheme \hat{R}^+ along with their theoretical values on 12 word pairs (Table 1) from a web crawl dataset.

Table 1: Information of 12 pairs of word vectors. Each word stands for a set of documents in which the word is contained. For example, “A” corresponds to the set of document IDs which contained word “A”.

Word 1	Word 2	f_1	f_2	R
HONG	KONG	940	948	0.925
RIGHTS	RESERVED	12,234	11,272	0.877
A	THE	39,063	42,754	0.644
UNITED	STATES	4,079	3,981	0.591
TOGO	GREENLAND	231	200	0.528
ANTILLES	ALBANIA	184	275	0.457
CREDIT	CARD	2,999	2,697	0.285
COSTA	RICO	773	611	0.234
LOW	PAY	2,936	2,828	0.112
VIRUSES	ANTIVIRUS	212	152	0.113
REVIEW	PAPER	3,197	1,944	0.078
FUNNIEST	ADDICT	68	77	0.028

Mean Square Error (MSE) of both estimators with respect to k which is the number of hash evaluations. To validate the theoretical variances (which is also the MSE because the estimators are unbiased), we also plot the values of the theoretical variances computed from Theorem 2 and Theorem 4. The results are summarized in Figure 6.

From the plots we can see that the theoretical and the empirical MSE values overlap in both the cases validating both Theorem 2 and Theorem 4. When k is small both the schemes have similar variances, but when k increases

the improved scheme always shows better variance. For very sparse pairs, we start seeing a significant difference in variance even for k as small as 100. For a sparse pair, e.g., “TOGO” and “GREENLAND”, the difference in variance, between the two schemes, is more compared to the dense pair “A” and “THE”. This is in agreement with Theorem 5.

7.2 Near Neighbor Retrieval with LSH

In this experiment, we evaluate the two hashing schemes \mathcal{H} and \mathcal{H}^+ on the standard (K, L) -parameterized LSH algorithm [14, 2] for retrieving near neighbors. Two publicly available sparse text datasets are described in Table 2.

Table 2: Dataset information.

Data	# dim	# nonzeros	# train	# query
RCV1	47,236	73	100,000	5,000
URL	3,231,961	115	90,000	5,000

In (K, L) -parameterized LSH algorithm for near neighbor search, we generate L different meta-hash functions. Each of these meta-hash functions is formed by concatenating K different hash values as

$$B_j(S) = [h_{j1}(S); h_{j2}(S); \dots; h_{jK}(S)], \quad (36)$$

where $h_{ij}, i \in \{1, 2, \dots, K\}, j \in \{1, 2, \dots, L\}$, are KL realizations of the hash function under consideration. The (K, L) -parameterized LSH works in two phases:

1. **Preprocessing Phase:** We construct L hash tables from the data by storing element S , in the train set, at location $B_j(S)$ in hash-table j .
2. **Query Phase:** Given a query Q , we report the union of all the points in the buckets $B_j(Q) \forall j \in \{1, 2, \dots, L\}$, where the union is over L hash tables.

For every dataset, based on the similarity levels, we chose a K based on standard recommendation. For this K we show results for a set of values of L depending on the recall values. Please refer to [2] for details on the implementation of LSH. Since both \mathcal{H} and \mathcal{H}^+ have the same collision probability, the choice of K and L is the same in both cases.

For every query point, the gold standard top 10 near neighbors from the training set are computed based on actual resemblance. We then compute the recall of these gold standard neighbors and the total number of points retrieved by the (K, L) bucketing scheme. We report the mean computed over all the points in the query set. Since the experiments involve randomization, the final results presented are averaged over 10 independent runs. The recalls and the points retrieved per query are summarized in Figure 7.

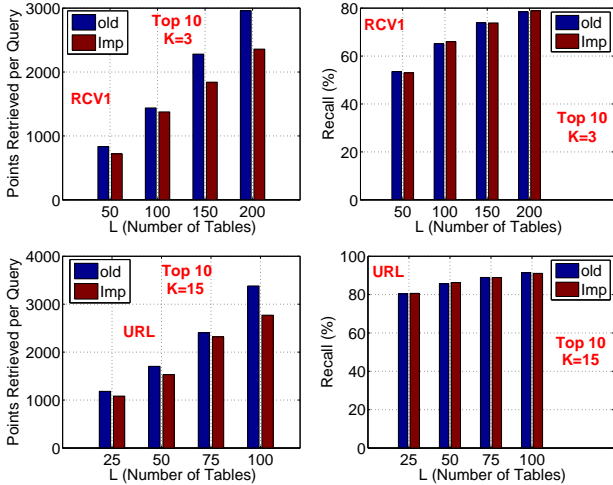


Figure 7: Average number of points scanned per query and the mean recall values of top 10 near neighbors, obtained from (K, L) -parameterized LSH algorithm, using \mathcal{H} (old) and \mathcal{H}^+ (Imp). Both schemes achieve the same recall but \mathcal{H}^+ reports fewer points compared to \mathcal{H} . Results are averaged over 10 independent runs.

It is clear from Figure 7 that the improved hashing scheme \mathcal{H}^+ achieves the same recall but at the same time retrieves less number of points compared to the old scheme \mathcal{H} . To achieve 90% recall on URL dataset, the old scheme retrieves around 3300 points per query on an average while the improved scheme only needs to check around 2700 points per query. For RCV1 dataset, with $L = 200$ the old scheme retrieves around 3000 points and achieves a re-

call of 80%, while the same recall is achieved by the improved scheme after retrieving only about 2350 points per query. A good hash function provides a right balance between recall and number of points retrieved. In particular, a hash function which achieves a given recall and at the same time retrieves less number of points is desirable because it implies better precision. The above results clearly demonstrate the superiority of the indexing scheme with improved hash function \mathcal{H}^+ over the indexing scheme with \mathcal{H} .

7.3 Why \mathcal{H}^+ retrieves less number of points than \mathcal{H} ?

The number of points retrieved, by the (K, L) parameterized LSH algorithm, is directly related to the collision probability of the meta-hash function $B_j(\cdot)$ (Eq.(36)). Given S_1 and S_2 with resemblance R , the higher the probability of event $B_j(S_1) = B_j(S_2)$, under a hashing scheme, the more number of points will be retrieved per table.

The analysis of the variance (second moment) about the event $B_j(S_1) = B_j(S_2)$ under \mathcal{H}^+ and \mathcal{H} provides some reasonable insight. Recall that since both estimators under the two hashing schemes are unbiased, the analysis of the first moment does not provide information in this regard.

$$\begin{aligned} & \mathbb{E}[1\{\mathcal{H}_{j1}(S_1) = \mathcal{H}_{j1}(S_2)\} \times 1\{\mathcal{H}_{j2}(S_1) = \mathcal{H}_{j2}(S_2)\}] \\ &= \mathbb{E}[M_{j1}^N M_{j2}^N + M_{j1}^N M_{j2}^E + M_{j1}^E M_{j2}^N + M_{j1}^E M_{j2}^E] \end{aligned}$$

As we know from our analysis that the first three terms inside expectation, in the RHS of the above equation, behaves similarly for both \mathcal{H}^+ and \mathcal{H} . The fourth term $\mathbb{E}[M_{j1}^E M_{j2}^E]$ is likely to be smaller in case of \mathcal{H}^+ because of smaller values of p . We therefore see that \mathcal{H} retrieves more points than necessary as compared to \mathcal{H}^+ . The difference is visible when empty bins dominate and $M_1^E M_2^E = 1$ is more likely. This happens in the case of sparse datasets which are common in practice.

8 Conclusion

Analysis of the densification scheme for one permutation hashing, which reduces the processing time of minwise hashes, reveals a sub-optimality in the existing procedure. We provide a simple improved procedure which adds more randomness in the current densification technique leading to a provably better scheme, especially for very sparse datasets. The improvement comes without any compromise with the computation and only requires $O(d+k)$ (linear) cost for generating k hash evaluations. We hope that our improved scheme will be adopted in practice.

Acknowledgement

Anshumali Shrivastava is a Ph.D. student partially supported by NSF (DMS0808864, III1249316) and ONR (N00014-13-1-0764). The work of Ping Li is partially supported by AFOSR (FA9550-13-1-0137), ONR (N00014-13-1-0764), and NSF (III1360971, BIGDATA1419210).

A Proofs

For the analysis, it is sufficient to consider the configurations, of empty and non-empty bins, arising after throwing $|S_1 \cup S_2|$ balls uniformly into k bins with exactly m non-empty bins and $k-m$ empty bins. Under uniform throwing of balls, any ordering of m non-empty and $k-m$ empty bins is equally likely. The proofs involve elementary combinatorial arguments of counting configurations.

A.1 Proof of Lemma 1

Given exactly m simultaneously non-empty bins, any two of them can be chosen in $m(m-1)$ ways (with ordering of i and j). Each term $M_i^N M_j^N$, for both simultaneously non-empty i and j , is 1 with probability $R\tilde{R}$ (Note, $\mathbb{E}(M_i^N M_j^N | i \neq j, I_{emp}^i = 0, I_{emp}^j = 0) = R\tilde{R}$).

A.2 Proof of Lemma 2

The permutation is random and any sequence of simultaneously m non-empty and remaining $k-m$ empty bins are equal likely. This is because, while randomly throwing $|S_1 \cup S_2|$ balls into k bins with exactly m non-empty bins every sequence of simultaneously empty and non-empty bins has equal probability. Given m , there are total $2m(k-m)$ different pairs of empty and non-empty bins (including the ordering). Now, for every simultaneously empty bin j , i.e., $I_{emp}^j = 1$, M_j^E replicates M_t^N corresponding to nearest non-empty Bin t which is towards the circular right. There are two cases we need to consider:

Case 1: $t = i$, which has probability $\frac{1}{m}$ and

$$\mathbb{E}(M_i^N M_j^E | I_{emp}^i = 0, I_{emp}^j = 1) = \mathbb{E}(M_i^N | I_{emp}^i = 0) = R$$

Case 2: $t \neq i$, which has probability $\frac{m-1}{m}$ and

$$\begin{aligned} & \mathbb{E}(M_i^N M_j^E | I_{emp}^i = 0, I_{emp}^j = 1) \\ &= \mathbb{E}(M_i^N M_t^N | t \neq i, I_{emp}^i = 0, I_{emp}^t = 0) = R\tilde{R} \end{aligned}$$

Thus, the value of $\mathbb{E}\left[\sum_{i \neq j} M_i^N M_j^E \middle| m\right]$ comes out to be

$$2m(k-m) \left[\frac{R}{m} + \frac{(m-1)R\tilde{R}}{m} \right]$$

which is the desired expression.

A.3 Proof of Lemma 3

Given m , we have $(k-m)(k-m-1)$ different pairs of simultaneous non-empty bins. There are two cases, if the closest simultaneous non-empty bins towards their circular right are identical, then for such i and j , $M_i^E M_j^E = 1$ with probability R , else $M_i^E M_j^E = 1$ with probability $R\tilde{R}$. Let p be the probability that two simultaneously empty

bins i and j have the same closest bin on the right. Then $\mathbb{E}\left[\sum_{i \neq j} M_i^E M_j^E \middle| m\right]$ is given by

$$(k-m)(k-m-1) [pR + (1-p)R\tilde{R}] \quad (37)$$

because with probability $(1-p)$, it uses estimators from different simultaneous non-empty bins and in that case the $M_i^E M_j^E = 1$ with probability $R\tilde{R}$.

Consider Figure 3, where we have 3 simultaneous non-empty bins, i.e., $m = 3$ (shown by colored boxes). Given any two simultaneous empty bins Bin i and Bin j (out of total $k-m$) they will occupy any of the $m+1 = 4$ blank positions. The arrow shows the chosen non-empty bins for filling the empty bins. There are $(m+1)^2 + (m+1) = (m+1)(m+2)$ different ways of fitting two simultaneous non-empty bins i and j between m non-empty bins. Note, if both i and j go to the same blank position they can be permuted. This adds extra term $(m+1)$.

If both i and j choose the same blank space or the first and the last blank space, then both the simultaneous empty bins, Bin i and Bin j , corresponds to the same non-empty bin. The number of ways in which this happens is $2(m+1) + 2 = 2(m+2)$. So, we have

$$p = \frac{2(m+2)}{(m+1)(m+2)} = \frac{2}{m+1}.$$

Substituting p in Eq.(37) leads to the desired expression.

A.4 Proof of Lemma 4

Similar to the proof of Lemma 3, we need to compute p which is the probability that two simultaneously empty bins, Bin i and Bin j , use information from the same bin. As argued before, the total number of positions for any two simultaneously empty bins i and j , given m simultaneously non-empty bins is $(m+1)(m+2)$. Consider Figure 4, under the improved scheme, if both Bin i and Bin j choose the same blank position then they choose the same simultaneously non-empty bin with probability $\frac{1}{2}$. If Bin i and Bin j choose consecutive positions (e.g., position 2 and position 3) then they choose the same simultaneously non-empty bin (Bin b) with probability $\frac{1}{4}$. There are several boundary cases to consider too. Accumulating the terms leads to

$$p = \frac{\frac{2(m+2)}{2} + \frac{2m+4}{4}}{(m+1)(m+2)} = \frac{1.5}{m+1}.$$

Substituting p in Eq.(37) yields the desired result.

Note that $m = 1$ (an event with almost zero probability) leads to the value of $p = 1$. We ignore this case because it unnecessarily complicates the final expressions. $m = 1$ can be easily handled and does not affect the final conclusion.

References

- [1] A. Agarwal, O. Chapelle, M. Dudik, and J. Langford. A reliable effective terascale linear learning system. Technical report, arXiv:1110.4198, 2011.
- [2] A. Andoni and P. Indyk. E2lsh: Exact euclidean locality sensitive hashing. Technical report, 2004.
- [3] R. J. Bayardo, Y. Ma, and R. Srikant. Scaling up all pairs similarity search. In *WWW*, pages 131–140, 2007.
- [4] A. Z. Broder. On the resemblance and containment of documents. In *the Compression and Complexity of Sequences*, pages 21–29, Positano, Italy, 1997.
- [5] A. Z. Broder, M. Charikar, A. M. Frieze, and M. Mitzenmacher. Min-wise independent permutations. In *STOC*, pages 327–336, Dallas, TX, 1998.
- [6] G. Buehrer and K. Chellapilla. A scalable pattern mining approach to web graph compression with communities. In *WSDM*, pages 95–106, Stanford, CA, 2008.
- [7] J. L. Carter and M. N. Wegman. Universal classes of hash functions. In *STOC*, pages 106–112, 1977.
- [8] T. Chandra, E. Ie, K. Goldman, T. L. Llinares, J. McFadden, F. Pereira, J. Redstone, T. Shaked, and Y. Singer. Sibyl: a system for large scale machine learning.
- [9] M. S. Charikar. Similarity estimation techniques from rounding algorithms. In *STOC*, pages 380–388, Montreal, Quebec, Canada, 2002.
- [10] S. Chien and N. Immorlica. Semantic similarity between search engine queries using temporal correlation. In *WWW*, pages 2–11, 2005.
- [11] F. Chierichetti, R. Kumar, S. Lattanzi, M. Mitzenmacher, A. Panconesi, and P. Raghavan. On compressing social networks. In *KDD*, pages 219–228, Paris, France, 2009.
- [12] D. Fetterly, M. Manasse, M. Najork, and J. L. Wiener. A large-scale study of the evolution of web pages. In *WWW*, pages 669–678, Budapest, Hungary, 2003.
- [13] M. R. Henzinger. Finding near-duplicate web pages: a large-scale evaluation of algorithms. In *SIGIR*, pages 284–291, 2006.
- [14] P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *STOC*, pages 604–613, Dallas, TX, 1998.
- [15] P. Li and K. W. Church. Using sketches to estimate associations. In *HLT/EMNLP*, pages 708–715, Vancouver, BC, Canada, 2005.
- [16] P. Li, K. W. Church, and T. J. Hastie. Conditional random sampling: A sketch-based sampling technique for sparse data. In *NIPS*, pages 873–880, Vancouver, BC, Canada, 2006.
- [17] P. Li, A. C. König, and W. Gui. b-bit minwise hashing for estimating three-way similarities. In *Advances in Neural Information Processing Systems*, Vancouver, BC, 2010.
- [18] P. Li, A. B. Owen, and C.-H. Zhang. One permutation hashing. In *NIPS*, Lake Tahoe, NV, 2012.
- [19] P. Li, A. Shrivastava, J. Moore, and A. C. König. Hashing algorithms for large-scale learning. In *NIPS*, Granada, Spain, 2011.
- [20] M. Mitzenmacher and S. Vadhan. Why simple hash functions work: exploiting the entropy in a data stream. In *SODA*, 2008.
- [21] M. Najork, S. Gollapudi, and R. Panigrahy. Less is more: sampling the neighborhood graph makes salsa better and faster. In *WSDM*, pages 242–251, Barcelona, Spain, 2009.
- [22] N. Nisan. Pseudorandom generators for space-bounded computations. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, STOC, pages 204–212, 1990.
- [23] A. Shrivastava and P. Li. Beyond pairwise: Provably fast algorithms for approximate k-way similarity search. In *NIPS*, Lake Tahoe, NV, 2013.
- [24] A. Shrivastava and P. Li. Densifying one permutation hashing via rotation for fast near neighbor search. In *ICML*, Beijing, China, 2014.
- [25] S. Tong. Lessons learned developing a practical large scale machine learning system. <http://googleresearch.blogspot.com/2010/04/lessons-learned-developing-practical.html>, 2008.
- [26] K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg. Feature hashing for large scale multitask learning. In *ICML*, pages 1113–1120, 2009.

First-Order Open-Universe POMDPs

Siddharth Srivastava and Stuart Russell and Paul Ruan and Xiang Cheng

Computer Science Division

University of California

Berkeley, CA 94720

Abstract

Open-universe probability models, representable by a variety of probabilistic programming languages (PPLs), handle uncertainty over the existence and identity of objects—forms of uncertainty occurring in many real-world situations. We examine the problem of extending a declarative PPL to define decision problems (specifically, POMDPs) and identify non-trivial representational issues in describing an agent’s capability for observation and action—issues that were avoided in previous work only by making strong and restrictive assumptions. We present semantic definitions that lead to POMDP specifications provably consistent with the sensor and actuator capabilities of the agent. We also describe a variant of point-based value iteration for solving open-universe POMDPs. Thus, we handle cases—such as seeing a new object and picking it up—that could not previously be represented or solved.

1 INTRODUCTION

The development of probabilistic programming languages or PPLs (Koller, McAllester, and Pfeffer, 1997; Milch et al., 2005; Goodman et al., 2008) has greatly expanded the expressive power of formal representations for probability models. In particular, PPLs express *open-universe probability models*, or OUPMs, which allow uncertainty over the existence and identity of objects. OUPMs are a natural fit to tasks such as vision, natural language understanding, surveillance, and security, where the set of relevant objects is not known in advance and the observations (pixels, strings, radar blips, login names) do not uniquely identify the entities in question.

It is natural, therefore, to consider whether the same benefits can be obtained for decision models, thereby providing a broader foundation for rational agents. A general decision

model—a partially observable Markov decision process or POMDP—includes a specification of what the agent can do and what it will perceive in any given state, as well as the reward functions.

For less expressive languages such as Bayesian networks and closed-universe first-order languages, the extension from probability models to decision models is relatively straightforward. For open-universe models, however, there are significant difficulties in defining POMDP representations that are both expressive enough to model the real world and mean exactly what is intended. When sensors supply sentences about named objects and actuators receive commands to act on named objects, problems arise if the formal names have uncertain referents. Consider the following example:

The sensors of an airport security system include passport scanners at check-in kiosks, boarding pass scanners, X-ray scanners, etc. A person passing through the airport generates observations from each of these scanners. Thus, the passport scanner at location A may generate observations of the form $IDName(p_{A,1}) = \text{“Bond”}$, $IDNumber(p_{A,1}) = 174666007$, $HeightOnID(p_{A,1}) = 185cm$, ...; a boarding-pass scanner at B may generate a sequence of the form $Destination(p_{B,7}) = \text{“Paris”}$, $IDNumber(p_{B,7}) = 174666007$, and finally, an X-ray scanner at C may generate observations of the form $MeasuredHeight(p_{C,32}) = 171cm$, $MeasuredHeight(p_{C,33}) = 183cm$.

In these observation streams, the symbols $p_{A,i}$, $p_{B,j}$ and $p_{C,k}$ are place-holder identifiers (essentially Skolem constants or “gensyms” in Lisp terminology). Although each use of a given symbol necessarily corresponds to the same individual, different symbols may or may not correspond to different individuals; thus, it is possible that $p_{A,1}$ and $p_{C,32}$ refer to the same person, while it is also possible that $p_{A,1}$ and $p_{B,7}$ refer to different people even though they are carrying documents with the same ID number.

The problems in modeling such a scenario are not limited to probabilistic models and can be illustrated using first-order

logic alone. For the observation model, we might want to say that, “Everyone in the security line will get scanned”:

$$\forall x \text{ InLine}(x) \rightarrow \text{Scanned}(x)$$

and “For everyone who gets scanned, we will observe a measured height”:

$$\forall x \text{ Scanned}(x) \rightarrow \text{Observable}(\text{MeasuredHeight}(x)). \quad (1)$$

So far, so good. Now, suppose we know, “Bond and his fiancée are in the security line.” While it is true, in a sense, that we will get a measured height for Bond’s fiancée, it is *not* true that the X-ray scanner will tell us:

$$\text{MeasuredHeight}(\text{Fiancée}(\text{Bond})) = 171\text{cm}.$$

Technically, the problem arises because we are trying to substitute $\text{Fiancée}(\text{Bond})$ for x in the universally quantified sentence (1), but one occurrence of x is inside $\text{Observable}(\cdot)$, which is a *modal operator*; thus, we have a failure of referential transparency—perhaps not surprising as we are trying to model what the agent will come to know. Practically, the problem arises because the sensor doesn’t know who Bond’s fiancée is. The same issue can arise on the action side: telling the security guard to “Arrest Bond’s fiancée” doesn’t work if the guard doesn’t know who Bond’s fiancée is. Thus, any proposed formal approach has to provide solutions to three fundamental problems: how to incorporate detected objects in an agent’s belief while allowing identity uncertainty; how to accurately state what can be sensed, and how to ensure that arguments in commands refer to something meaningful for actuators.

In fact, communication between the physical layer and the formal model requires a restricted vocabulary whose terms are guaranteed to be meaningful. This is because in a probability model, an observation has to be *true* to be conditioned on. The problem is that an OUPOMDP framework needs to ensure this even when the observations use uncertain references. We formally define the terms that will be meaningful, and therefore suitable for use in communication with the physical layer, using the concept of *rigid designators* from modal logic. This notion facilitates a framework where observations are true statements, without being restrictive. Our solution is analogous to modeling uncertain observations of a property as certain observations about a noisy version of that property.

We begin in §2.1 by showing, as a “warm-up exercise,” how POMDPs may be defined on a substrate of dynamic Bayesian networks (DBNs). §2.2 provides additional background on first-order, open-universe probability models. §3 describes our proposed semantics for a decision-theoretic extension of the BLOG language. We show that the resulting framework models sensor and actuator specifications accurately. §4 describes a variant of the point-based value iteration (PBVI) algorithm designed to handle open-universe POMDPs, and §5 describes experiments in a simple domain that illustrate the ability of the formalism and

algorithm to handle problems that previous formalisms (§6) cannot express and previous algorithms cannot solve. It has been possible for many years to program a robot to walk into a room, see something, and pick it up; now, it is possible for the robot to do this without leaving the formal framework of rational decision making.

2 BACKGROUND

2.1 POMDPs

A POMDP defines a decision-theoretic planning problem for an agent. At every step, the agent executes an action, then the environment enters a new state, then the agent receives an observation and a reward.

Definition 1. A POMDP is defined as $\langle X, U, O, T, \Omega, R, \gamma \rangle$, where X, U, O are finite sets of states, actions and observations; $T(X_{t+1} = x' \mid X_t = x, U_t = u)$ defines the transition model, i.e., the probability of reaching state x' if action u is applied in state x ; $\Omega(O_{t+1} = o \mid X_{t+1} = x', U_t = u)$ defines the observation model, i.e., the probability of receiving an observation o when state x' is reached via action u ; and $R : X \times U \rightarrow \mathbb{R}$ defines the reward that the agent receives on applying a given action at a given state.

The agent’s belief state at any timestep is the probability distribution over X at that timestep, given the initial belief and the history of executed actions and obtained observations. POMDP solutions typically map observation histories to actions (Kaelbling, Littman, and Cassandra, 1998). We will refer to such solutions as *observation-history policies*. On the other hand, *belief-state policies* map the agent’s belief states to actions. Both solution representations are equally expressive because the belief state constitutes a sufficient statistic for the agent’s history and the initial belief (Astrom, 1965). An optimal POMDP policy maximizes the expected value of the aggregate reward $\sum_{i=1}^{\infty} \gamma^i \cdot r(i)$ where $r(i)$ is the reward obtained at timestep i and $\gamma \leq 1$.

A DBN (Dean and Kanazawa, 1989) describes a factored, homogeneous, order-1 Markov process as a “two-slice” Bayesian network showing how variables at time $t + 1$ depend on variables at t . At each timestep, any subset of variables may be observed. To represent POMDPs, Russell and Norvig (1995) assume a fixed set of always-observable evidence nodes and define *dynamic decision networks* (DDNs) by adding to each timestep of a DBN the following variables:

- An *action variable* U_t whose values are the possible actions at time t . For now, assume this set of actions is fixed. The POMDP’s transition model is represented through the conditional dependence of variables at $t + 1$ on the value of U_t and other variables at t .
- A reward variable R_t , which depends deterministi-

cally on U_t and other variables at time t .

This definition of POMDPs by DDNs is not, however, completely general, since it does not allow for the observability of a variable to depend on the state. Clearly, different observability conditions correspond to different POMDPs. To handle state-dependent observability, one can define for each variable X_i that may be observed, a second Boolean variable $ObsX_i$ that captures whether or not X_i is observed¹. This factors out dependencies for observability from dependencies for the variable values. Thus, in order to define POMDPs we can define DDNs consisting of the following nodes in addition to the action and reward nodes:

- A Boolean *observability variable* $ObsX_i$ for each ordinary variable X_i . Each observability variable is necessarily observed at each time step, and $ObsX_i$ is true iff X_i is observed. Observability variables may have as parents other observability variables, ordinary variables, or the preceding action variable. The DBN with X_i and $ObsX_i$ variables defines Ω .

In this model, an always-observable X_i has an $ObsX_i$ with a deterministic prior set to 1.

In order to define first-order OUPOMDPs, we need to extend first-order OUPMs in a manner analogous to the DDN extension of DBNs. However, as we will show below, the analogous extension leads to conflicts with what may be known to the agent during decision making, and results in the models of sensors and actuators with unintentionally broad capabilities as seen in the introduction.

2.2 OPEN-UNIVERSE PROBABILITY MODELS

First-Order Vocabularies Given a set of types $\mathcal{T} = \{\tau_1, \dots, \tau_k\}$, a first-order *vocabulary* is a set of function symbols with their type signatures. Constant symbols are viewed as zero-ary function symbols and predicates as Boolean functions. A *possible world* is defined as a tuple $\langle \mathcal{U}, \mathcal{I} \rangle$ where the universe $\mathcal{U} = \langle \mathcal{U}_1, \dots, \mathcal{U}_k \rangle$ and each \mathcal{U}_i is a set of elements of type $\tau_i \in \mathcal{T}$. The interpretation \mathcal{I} has, for each function symbol in the vocabulary, a function of the corresponding type signature over $\mathcal{U}_1, \dots, \mathcal{U}_k$. The set of types includes the type *Timestep*, whose elements are the natural numbers. Functions whose last argument is of type timestep are called *fluents*. A state is defined by the values of all static functions and fluents at a particular timestep in a possible world.

¹Observability variables capture the full range of possibilities in the spectrum from missing-completely-at-random (MCAR) to not-missing-at-random (NMAR) data (Little and Rubin, 2002). An alternative would be to say that “null” values are observed when a variable is not observable. However, this approach has distinct disadvantages as it requires (a) unnecessarily large parent sets for evidence variables capturing when null values may be obtained, and (b) additional mechanisms for handling dependencies of child nodes of variables that may get a null value.

```

1 Type Urn, Ball;
2 origin Urn Source(Ball);
3 #Urn ~ Poisson(5);
4 #Ball(Source = u) {
5   if Large(u) then ~ Poisson(10)
6   else ~ Poisson(2)};
7 random Boolean Large(Urn u)
8   ~ Categorical{true->0.5, false->0.5};

```

Figure 1: A BLOG model illustrating number statements.

Open-Universe Probability Models in BLOG Our approach can be applied to formal languages for generative, open-universe probability models (OUPMs). BLOG (Milch et al., 2005; Milch, 2006) is one such language. We refer the reader to the cited references for details on this system, and discuss briefly the components relevant to this paper. A BLOG model consists of two types of statements: (1) number statements, which specify conditional probability distributions (cpds) for the number of objects of each type in the universe of a structure; and (2) dependency statements, which specify cpds for the values of functions applied on the elements of the universe.

Each type can have multiple number statements and each number statement can take other objects as arguments. Fig. 1 shows a simple example of a BLOG model with two types, *Urn* and *Ball*. This model expresses a distribution over possible worlds consisting of varying numbers of urns with varying numbers of balls in each urn. The number of urns follows a Poisson(5) distribution (line 3). The number of balls in an urn depends on whether or not the urn is *Large*. *Origin* functions map the object being generated to the arguments that were used in the number statement that was responsible for generating it. In Fig. 1, *Source* maps a ball to the urn it belongs to. The number of balls in an urn follows a Poisson(10) distribution if the urn is *Large*, and a Poisson(2) distribution otherwise (lines 4-6). Finally, the probability of an urn being *Large* is 0.5 (lines 7 & 8).

Evidence statements in BLOG provide first-order sentences in the model’s vocabulary as observations; e.g.,

obs (exists Ball b exists Urn u Source(b)==u & Large(u)) = true
states that there exists a large urn that has a ball.

A *set evidence* statement provides a concise syntax for naming all the distinct objects satisfying a certain predication; e.g.,

obs {Ball b: exists Urn u Source(b)==u & !Large(u)}={b1, b2}

states that the set of balls in small urns consists of exactly two balls, referred to by the constant symbols *b1* and *b2*.

We denote the execution of an action $u(x)$ at a timestep t using the fluent $apply_u(\bar{x}, t)$. The effects of an action are represented by defining the value of all fluents affected by the action at timestep $t + 1$ in terms of the fluents and non-

fluents at timestep t . The reward function can be expressed as a fluent in a similar manner. This notation is similar to successor-state axioms in situation calculus (Reiter, 2001). For example, a *sendToScanner*(p, t) action may result in the person p going to the scanner. Let *followedInstructionCPD*, *leftScannerCPD* and *defaultScannerCPD* denote respectively the probability distribution that a person follows instructions, that s/he has left the scanner and that s/he is already at the scanner. We express the effect of this action on the predicate *atScanner* using the fluent *apply_sendToScanner*(x, t) as follows:

```
atScanner(Person p, Timestep t+1) {
  if apply_sendToScanner(p,t)
    then ~ followedInstructionCPD()
  else if atScanner(p, t) then ~ leftScannerCPD()
  else ~ defaultAtScannerCPD();
```

Representation theorems for BLOG ensure that every well-founded specification (that does not create cyclic dependency statements or infinite ancestor sets) corresponds to a unique probability distribution over all possible worlds.

3 DECISION-THEORETIC BLOG

In this section we present the key components of decision-theoretic BLOG (DTBLOG), which adds to the BLOG language decision variables for representing actions and meta-predicates for representing observability.

3.1 SOLUTION APPROACH

A natural generalization of the *ObsX_i* idea discussed in §2.1 is to write dependencies of the form: *Observable*($\psi(\bar{x})$) {if $\varphi(\bar{x})$ then $\sim cpd_1$ }, where w.l.o.g., φ and ψ can be considered to be predicates defined using FOL formulas with variables in \bar{x} as free variables. The interpretation of this formula would be “ $\psi(\bar{x})$ is observed with probability given by cpd_1 when $\varphi(\bar{x})$ holds”. As noted in the introduction, this formulation leads to problems associated with *referential transparency* in first-order reasoning. In order to model the sensor accurately, we want to describe a belief state where *Vesper* = *Fiancee*(*Bond*), *Scanned*(*Vesper*) = *Scanned*(*Fiancee*(*Bond*)) = true, *Observable*(*MeasuredHeight*(*Vesper*)) = true, but *Observable*(*MeasuredHeight*(*Fiancee*(*Bond*))) = false.

However, these statements will be inconsistent in any system based on first-order reasoning. This is because first-order logic enforces all terms to be referentially transparent, as a consequence of the application of the axiom of universal instantiation on the substitution axioms for equality. The axiom of universal instantiation states that if $\forall x \alpha(x)$ is true, then for any ground term t , $\alpha(t/x)$ (the version of $\alpha(x)$ with all free occurrences of x replaced by t) is also true. Different truth values for the observability statements above stem from the fact that the sensor knows who *Vesper* is, but not who *Fiancee*(*Bond*) is. This indicates that the knowledge of sensors and actuators needs

to be taken into account while determining the possible effects of communicating with them.

To address this problem, we draw upon modal logic, where knowledge of the agent is taken into account (Levesque and Lakemeyer, 2000). The modal logic version of the axiom of universal instantiation states that if $\forall x \alpha(x)$ is true, then for any ground term t whose value is known, $\alpha(t/x)$ holds. In order to determine which terms are known, modal logic develops the notion of *rigid designators*: terms that have unique interpretations in all possible worlds according to the agent’s knowledge. Our formulation ensures that only the modal logic version of the axiom of universal instantiation is used with statements involving observability and doability. In practice, we use a modal logic of observations, so that a term is considered to be known to a system iff its value is predefined or has been observed. Our framework to ensure that (a) observations are true statements and are thus suitable for conditioning, and (b) terms used in the interface with actuators have well-defined values.

3.2 FORMAL SPECIFICATIONS

We begin our formal solution with elementary descriptions of physical sensors and actuators that clarify the types of rigid designators that sensors (actuators) can provide (act upon). We assume that the vocabulary always includes, as rigid designators, constant symbols for elements of standard types such as natural numbers and strings; e.g., the constant symbol 1 represents the natural number 1 in every possible world. We also assume that standard mathematical and string operations have fixed interpretations and that terms that are applications of fixed functions on rigid designators are therefore also rigid designators (e.g., $1 + 2$). As explained in the following sections, observations may add to the set of rigid designators.

Any physical sensor can be described in terms of the types of symbols it can generate and the type signatures of the properties that it can report:

Definition 2. A sensor specification S is a tuple $\langle \bar{T}_S, \tau_S \rangle$ where \bar{T}_S is a tuple of types for the observation values that S produces and τ_S denotes the type of new symbols that S may generate.

E.g., an X-ray scanner can be specified as: *scanner* = $\langle \langle \text{PersonRef}, \text{Length} \rangle, \langle \text{PersonRef} \rangle \rangle$. Such a scanner can generate new symbols of type *PersonRef* and returns observation tuples of the type $\langle \text{PersonRef}, \text{Length} \rangle$; elements of type *Length* are real numbers with units (not terms such as *Height*(*Bond*)). Note that such a specification indicates that rigid designators of type \bar{T}_S and τ_s are meaningful to the sensor, but arbitrary terms of the same types may not be.

A physical actuator can be specified similarly:

Definition 3. An actuator specification A is a tuple of types

\bar{T}_A denoting the types of its arguments.

E.g., an actuated camera may be able to take a picture given an orientation and focusing distance: $TakePhoto = \langle Orientation, Length \rangle$.

Let B be a set of beliefs, U be a set of actions, and O be a set of observations. A *strategy tree* $T_{B,U,O} = \langle \ell_U, \ell_O \rangle$ is defined by a mapping $\ell_U : B \rightarrow 2^U$, which defines the set of possible actions at each belief state, and a mapping $\ell_O : B, U \rightarrow 2^O$, which defines the set of possible observations when an action is applied on a belief state. Given a set of sensors \mathcal{S} and actuators \mathcal{A} , an *actuator mapping* $\alpha_A : U \rightarrow \mathcal{A}$ denotes the actuator responsible for executing each $u \in U$ and a *sensor mapping* $\sigma_S : O \rightarrow \mathcal{S}$ denotes the sensor responsible for generating o . We drop the subscripts \mathcal{S} and \mathcal{A} when they are clear from context.

Definition 4. A term is a rigid designator in a belief state b if it has a unique value in all possible worlds that have nonzero probability (in a discrete setting) or nonzero density (in a continuous setting) in b .

Definition 5. A strategy tree $T_{B,U,O} = \langle \ell_U, \ell_O \rangle$ is well-defined w.r.t. a set of sensor specifications \mathcal{S} , actuator specifications \mathcal{A} , a sensor mapping σ , and an actuator mapping α iff for every $b \in B$:

- Every $u \in \ell_U(b)$ uses as terms only the rigid designators of type \bar{T}_A in b , where $A = \alpha(u)$.
- For every $u \in \ell_U(b)$, every $o \in \ell_O(b, u)$ is either the observation of symbol of type τ_S or uses as terms only the rigid designators of type \bar{T}_S in b , where $S = \sigma(o)$.

In other words, a strategy tree is well-defined iff the actions and observations that it specifies at each step are truly possible. A framework for solving OUPOMDPs therefore needs to ensure that the strategy trees it defines and uses in its solution algorithms are well-defined w.r.t. the given sensors and actuators. We now present our formulation of sensors and actuators for achieving this objective.

3.3 MODELING SENSORS

In this section we describe our formulation for a sensor $S = \langle \bar{T}_S, \tau_S \rangle$. Recall that τ_S is the set of symbols generated by S . By definition, such symbols evaluate to themselves (just like integers) and are therefore rigid designators. We specify S using the following components in DTBLOG:

- A predicate V_S with arguments \bar{T}_S, t , representing the tuples returned by the sensor.
- The statement $ObservableType(\tau_S)$; denoting that symbols of type τ_S are returned by the sensor. Number statements for τ_S constitute a generative model for the elements generated by S . Each number statement for symbols of type τ_S includes an origin function $\tau_S.time$ which maps the symbol to the time when it was generated.

```

0  #Person ~ Poisson[10]; LocKnownDuration=4;
1  #PersonRef(Src = p, PersonRef_Time=t) {
2    if AtScanner(t)=p ~Bernoulli(0.5)
3    else = 0;
4    observableType(PersonRef);
5    observable(MeasuredHt(p_ref, t))=
6      (AtScanner(t) == Src(p_ref));
7    MeasuredHt(p_ref, t) ~ Normal[Ht(Src(p_ref)), 5];
8    Ht(prsn) ~ Normal[160, 30];
9
10 decision apply_TakePhoto(Orientation o, Distance d,
11   Timestep t);
12 PictureTaken(Person p, Timestep t){
13   if exists d, o RelativeDistance(TrueLoc(p, t)) == d &
14     RelativeOrientation(TrueLoc(p, t)) == o &
15     apply_takePhoto(o, d) then = true
16   else = False;
17
18 //Entrance model
19 AtScanner(t) ~ UniformChoice({Person prsn:
20   !Entered(prsn, t)});
21 Entered(prsn, t){
22   if t>0 & (AtScanner(t-1)=prsn) then = true
23   elseif t>0 then = Entered(prsn, t-1)
24   elseif t=0 then = false;
25   TrueLoc(Person p, t) ~ MovementModel(TrueLoc(p, t-1));
26   Location(p_ref, t) {
27     if PersonRef_Time(p_ref) < Horizon+1 &
28       & t<= PersonRef_Time(p_ref)+LocKnownDuration
29     then ~ TrueLoc(Src(p_ref, t))
30     else = null;

```

Figure 2: A DTBLOG model for the airport domain

- A dependency for $Observable(V_S(\bar{T}_S, Timestep))$, indicating the factors influencing the probability that S generates an observation.

For ease in representation, we also allow syntax for capturing sensors that return values of functions declared in the model's vocabulary, rather than the default V_S predicate. Such observations also create rigid designators. For instance, it may be convenient to represent the scanner as a sensor that provides values of the measured height, captured by the function $MeasuredHt_{scanner}(p, t)$. In this case, the relational observability statements would provide dependencies for $Observable(MeasuredHt_{scanner}(p, t))$. Lines 4-6 in Fig.2 capture the DTBLOG specification for such a scanner. If the agent receives an observation of the form $MeasuredHt_{scanner}(pref_{10,1}, 7) = 171cm$, then $MeasuredHt_{scanner}(pref_{10,1}, 7)$ gets a unique interpretation, and thus becomes a rigid designator.

3.3.1 GENERATION OF OBSERVATIONS

Intuitively, if the environment is in state q at timestep t , and $Observable(\varphi(\bar{x}))$ (or $ObservableType(\tau_S)$) is true in q , then the value of $\varphi(\bar{x})$ (or the symbols of type τ_S generated at a timestep t) must be obtained as evidence at timestep t . We operationalize this intuition through two types of observations in DTBLOG: observations of the symbols generated by sensors, and observations about properties of the observed symbols.

Symbol observations All elements generated via a sensor's symbol observability statement are assigned unique names and provided to the agent as an evidence statement. Suppose the model includes an observability declaration of

the form $ObservableType(\tau_S)$. For every possible world w at timestep t , this declaration results in the generation of a set evidence statement of the following form:

obs $\{\tau_S \ c:\tau_S_time(c) == t\} = \{c_{t,1}, \dots, c_{t,k(t)}\}$;

where $k(t) = \#\{\tau_S \ x : \tau_S_time(x) == t\}$ in w . Such a set evidence statement models the observation of a set of symbols generated by the sensor S . The semantics of BLOG ensure that $c_1, \dots, c_{k(t)}$ are interpreted as distinct objects; e.g., the scanner declaration in Fig. 2 results in the generation of a set of *PersonRef* objects at each timestep. The number of such symbols is given by the CPD for the number statement for *PersonRef*. At timestep 10, this observation could be:

obs $\{PersonRef \ x: PersonRef_Time(x) == 10\} = \{pref_{10,1}\}$;

Relational observations For every true $Observable(\varphi(\bar{x}))$ atom in a state, the DTBLOG engine creates an observation statement of the form:

obs $\varphi(\bar{x}) = v$;

where all arguments and the value v are rigid designators. E.g., if in a possible world $Observable(MeasuredHt_scanner(pref_{10,1}, 10)) = true$ and $MeasuredHt_scanner(pref_{10,1}, 10) = 171cm$, the generated relational observation would be:

obs $MeasuredHt_scanner(pref_{10,1}, 10) = 171cm$;

Each argument in such evidence statements has to be a rigid designator. Returning to the informal example described in the introduction, suppose Bond and Vesper were generated *PersonRefs*. The DTBLOG engine will not generate an observation of the form $MeasuredHeight(Fiancee(Bond), t) = 150cm$ if the value of $Fiancee(Bond)$ has not been observed (unless $Fiancee$ is a fixed function, which would be quite unusual), even when $MeasuredHeight()$ is observable for all *PersonRefs*. Thus, the agent being modeled will not “expect” an observation of the form “ $MeasuredHeight(Fiancee(Bond), t) = 150cm$ ”. On the other hand, if the agent receives an observation of the form $Fiancee(Bond) = Vesper$, the term on the left becomes a rigid designator and future user-provided observations and system-generated decisions may use the term $Fiancee(Bond)$.

We summarize our formulation of sensor models by noting that the set of all observations corresponding to a sensor S , $\sigma^{-1}(S)$, consists of the set evidence statements of type τ_S and value evidence statements for the predicate V_S (or its functional representation as noted above).

3.4 MODELING ACTUATORS

We represent an actuator $A = \bar{T}_A$ in a DTBLOG model as a decision function $apply_A(\bar{T}_A, t)$. Decision functions are declared using the keyword *decision*; e.g., an actuated camera $TakePhoto = \langle Orientation, Length \rangle$ can be specified as: decision $apply_TakePhoto(Orientation, Distance, Timestep)$;

The space of all possible actions in a belief state could be defined using all possible substitutions of the arguments of decision variables with terms. However, without any further restrictions, this would lead to unintended situations where the user provides a decision of the form:

apply_TakePhoto(Orientation(Loc(Src(pref_{10,1}),t)),
DistanceTo(Loc(Src(pref_{10,1}),t)),t)=true;

even when $Loc(Src(pref_{10,1}),t)$ has not been observed. Such a decision would not only be meaningless to the actuator, it can lead to “fake” solutions, e.g., if the desired objective was to determine the current location of the person who generated $pref_{10,1}$ and take their photo. We eliminate such situations by allowing only rigid designators as arguments of decision functions. Lines 10-16 in Fig. 2 capture the DTBLOG specification for a camera. The remainder of Fig. 2 completes the specification with dependencies for the persons’ locations and their movement through the scanner. Since a *PersonRef* is generated when the someone is at the scanner, the location (of the person) corresponding to a *PersonRef* is known for a brief period.

We summarize our formulation of actuator models by noting that the set of all decisions corresponding to an actuator A , $\alpha^{-1}(A)$, consists of decision functions of the form $apply_A(\bar{T}_A)$.

3.5 OUPOMDP DEFINED BY A DTBLOG MODEL

Let $M(\mathcal{S}, \mathcal{A})$ be a DTBLOG model defined using the sets \mathcal{S} and \mathcal{A} of sensors and actuators respectively. Let \mathcal{V}_M be the first-order vocabulary used in M . Then, M defines an OUPOMDP $\langle X, U, O, T, \Omega, R \rangle$ where the set of states X is the set of all timestep-indexed states corresponding to the possible worlds of vocabulary \mathcal{V}_M . U is the set of all instantiated decision functions corresponding to \mathcal{A} that are allowed in some state $x \in X$ and O is the set of all instantiated functions corresponding to sensor specifications in \mathcal{S} that are allowed in some state $x \in X$. The transition function T and observation function Ω are defined by the probabilistic dependency statements in M . This formulation does not place any requirements on distributions of the values of observations or the effects of actions. Every BLOG model must include dependencies for every declared function; the dependencies for the values of sensor-predicates and fluents affected by decisions can be defined to capture arbitrary physical processes.

Strategy tree generated by a DTBLOG model DTBLOG represents belief states using collections of sampled, possible states. BLOG’s existing sampling subroutines are used to sample possible worlds corresponding to the initial state specified in the BLOG model. The semantics developed above implicitly define the strategy tree for a DTBLOG model: the application of a decision on a belief updates each possible world in the belief to the next timestep using the stated dependencies. When the belief

state is updated w.r.t. to a decision, the DTBLOG engine generates the set of observations corresponding to each possible updated state.

The following results follow from the semantics above.

Lemma 1. *The semantics of DTBLOG ensure that in a belief b , (a) every generated observation statement o is either a set evidence statement for symbols of type $\tau_{\sigma(o)}$ or an observation statement utilizing only rigid designators of type specified by $\bar{T}_{\sigma(o)}$, and (b) every possible decision u uses only arguments that are rigid designator tuples of the type $\bar{T}_{\alpha(u)}$.*

Theorem 1. *Let $M(S, A)$ be a DTBLOG model defined using sensor and actuator specifications S and A respectively, such that σ_S and α_A are its sensor and actuator mappings. If M satisfies BLOG's requirements for well-defined probabilistic models then it generates a well-defined strategy tree w.r.t. S , A , σ_S , and α_A .*

3.6 MODELING REAL-WORLD SITUATIONS

We now show that various non-trivial modes of sensing and acting can be expressed easily in the presented framework.

Observations of composed functions Consider a human operator who reads the passport of a passenger immediately after she exits the height scanner, and reports the date of birth along with the person reference generated by the height scanner. Intuitively, the human operator provides observations of a function composition: $DOB(PassportID(Src(PersonRef)))$, where DOB maps a passport number to the date of birth mentioned within. This appears to be a problem since the use of function application terms is specifically disallowed in our framework unless each such term has been observed. However, the situation can be modeled by defining the human operator as a sensor $\langle \langle PersonRef, Date \rangle, \langle \rangle \rangle$, that reports observations about the derived function $DOB \circ PPOf-Src(x) = DOB(PassportID(Src(x)))$. In fact, our formulation correctly ensures that the agent will not expect an observation of $DOB(PassportID(Src(x)))$ from a sensor that reports the $DOB(y)$ function (perhaps a swipe-through scanner).

Actions on sensor-generated symbols In the real world, agents need to act upon objects that are detected through their sensors. Clearly, actuators can be specified to directly take inputs of the type generated by a sensor. Most commonly however, physical actuators such as a robot's gripper cannot act upon symbols. In such situations, sensor-generated symbols can be used in actions that can be compiled down to primitive actions respecting the semantics defined above. Consider a situation where the scanner reports the estimated *Location* of the person generating a person reference in addition to their measured heights. Let the functions *RelativeOrientation* and *CamDistance* map positions to the orientation and distance relative to the camera, respectively. We can define a camera action that takes a

snapshot given a *PersonRef* as follows:

```
apply_TakePhotoPRef(p_ref, t) :=
    apply_TakePhoto(RelativeOrientation(Location(p_ref)),
                    CamDistance(Location(p_ref)), t)
```

Every instance of *apply_TakePhotoPRef()* is compiled out into the primitive action *apply_TakePhoto()*.

Actuators that gain knowledge Although this framework cannot capture completely the beliefs of actuators or sensors with their own reasoning abilities, it can model actuators and sensors whose knowledge evolves. Consider an actuator that can dial phone numbers. Suppose this actuator models a human operator, who can read the phone book to obtain the phone number for any name. Our system correctly allows decisions such as *apply_Dial(phoneNumber("John"), t)* only if an observation of *PhoneNumber("John")* is received. However, they should also be allowed if the operator has been able to look up John's phone number in the phone book. The non-trivial part in designing such a system is to model the growth in knowledge of the operator.

Such a phone operator can be specified as $\langle String, Time \rangle$. The corresponding decision variable takes the form *apply_Dial(s, t)*. The restrictions developed above, on possibly substitutions for s , apply here. The effects of this action are determined by a predicate:

```
dialNumber(n, t) {
    if exists String s apply_Dial(s, t)
        and n == lookedUpNumber(s, t) then = true
    else = false};
```

where *lookedUpNumber(s, t)* is either the number represented by s if s is the string representation of a number, or, if s is a name, then it is the looked up number for that name according to a certain distribution:

```
lookedUpNumber(String s, Time t) {
    if strToNumber(s) != null then = strToNumber(s)
    else if exists t' < t operatorLookedUp(s, t')
        then ~ phoneBookCPD(s)
    else = null};
```

With this formulation, the application of decisions like *apply_dialNumber("John")* will result in a correctly updated belief state, contingent upon whether or not the operator performed a lookup.

4 SOLVING OUPOMDPS: OU-PBVI

In order to illustrate the efficacy of our framework, we developed and implemented an open-universe version of point-based value iteration (PBVI) (Pineau et al., 2003). OU-PBVI is designed to handle the main aspects of OUPOMDPS that prevent a direct application of POMDP algorithms: (a) belief states are infinite dimensional objects, and (b) the set of possible observations is unbounded. Thus, standard POMDP algorithms, which rely upon carrying out analytical backup operations using the domain's

transition function cannot be applied in general (although closed form backups for special cases may be possible). The following description focuses upon the key enhancements made to PBVI; a description of PBVI itself can be found in the cited reference.

PBVI restricts the Bellman backup of the value function for POMDPs to a finite set of sampled belief states, B . The $\|X\|$ -dimensional α vector for a policy represents its value function in terms of the expected value of following the policy for each state $x \in X$. PBVI approaches construct $t + 1$ step policies by computing the $t + 1$ step value function for each $b \in B$:

$$V^{t+1}(b) = \max_u \{E_b[r(s, u)] + \gamma E_{\Omega(o|b, u)}[V_{max}^t(b_o^u)]\},$$

where $V_{max}^t(b_o^u) = \max_{\alpha \in V_t} \alpha \cdot b_o^u$, and b_o^u is the belief state after action u is applied in b and observation o is obtained. $V^{t+1}(b)$ represents a 1-step backup of V for b .

In order to apply these ideas to OUPOMDPs, OU-PBVI represents belief states as sampled sets of possible worlds. Action application and the generation of observations proceeds as discussed in detail in the preceding sections. Thus, for each sampled state in a belief state b , action application results in a set of observations corresponding to the sampled states in b . We use a particle filter with N_p particles for belief propagation, and replace expectations by sample averages (Srivastava et al., 2012). In addition, we evaluate α vectors lazily. In our implementation, the number of keys (possible worlds) in α may reach a bound of $N_p \cdot \|B\|$. We also store the policy corresponding to each α vector. This allows us to dynamically compute missing components of the vector, as needed. More precisely, during backup, if the value of a policy π is required for a state that is not in the current set of states captured by α_π , that state is added to α_π and its value is computed by carrying out N_E simulations of the execution of that policy. During each simulation, after each step of policy execution and observation generation, if the resulting sampled state x is present in $\alpha_{\pi'}$ where π' is the subpolicy of π that remains to be executed, $\alpha_{\pi'}(x)$ is used and the simulation terminates. The estimates of policy value functions converge to the true value functions as the $N_p, N_E \rightarrow \infty$.

5 TEST PROBLEM AND RESULTS

We conducted experiments to investigate whether point-based approaches could be used for solving OUPOMDPs. As a test problem we developed an open-universe version of the Tiger problem. The agent is surrounded by 4 zones with an unknown, unbounded number of tigers who may move among zones at each timestep. Multiple tigers may be in a zone and the objective is to enter a zone without a tiger. The agent has two actions, a *listen*(*Timestep* t) action that allows it to make inaccurate observations about the growls made by tigers at timestep t , and an *enter*(*Zone*

z , *Timestep* t) action which it can use to enter a zone. When *listen* is applied, the agent obtains a growl from each tiger with probability 0.5.

observableType(Growl);

#Growl(Source = m, Time_Growl = t) {

 if apply_listen(t-1) then \sim Bernoulli(0.5)

 else = 0};

The listen action also gives a noisy estimate of the zones from which growls came. If a growl is made by a tiger in z , the probability of observing that a growl was made at z is 0.75, and that of observing that a growl was made at each of the zones $(z + 1 \bmod 4)$ and $(z - 1 \bmod 4)$ is 0.25. At each timestep, each tiger independently either stays in its zone with probability 0.4 or moves to each of the neighboring zones with probability 0.3. The agent receives a reward 10 for entering a door without tiger, -1 for listening, and -100 for entering a zone with a tiger. The number of tigers follows a Poisson(1) distribution. The objective is to find a policy for the belief state specified in the DTBLOG model. The unknown, *unbounded* numbers of tigers and *relevant* observations, as well as the independent movements of tigers make this a difficult OUPOMDP that cannot be expressed using existing approaches.

For our evaluation, we fixed $N_E = 100$, $\gamma = 0.9$ and the time horizon at 5. Fig. 3 shows the values of the obtained policies for varying values of $\|B\|$ and N_p , averaged over 10 different runs of the algorithm. To compute estimates of policy values, we carried out N_E simulations of the computed policy on each of 5000 samples for the initial state. The results show that the variance in policy value-estimates decrease as N_p and $\|B\|$ increase.

Since no other existing approach addresses OUPOMDPs, we used a *belief-state query* (BSQ) policy (Srivastava et al., 2012) as a baseline. BSQ policies map first-order probabilistic queries to belief states. Although such policies are more succinct than observation-history policies, evaluating them is non-trivial because the action to be applied at each step depends on the posterior probability of a first-order query rather than just the observation history. Algorithms for policy evaluation are discussed in detail in the cited reference. For our experiments we used the following, intuitively simple BSQ policy:

if $\Pr(\text{no tiger behind } d_1 \text{ at } t) > \theta$, *enter*(d_1, t)

else if $\Pr(\text{no tiger behind } d_2 \text{ at } t) > \theta$, *enter*(d_2, t)

...

else *listen*(t)

We found that $\theta = 0.9$ resulted in the highest expected value for this BSQ policy, shown using the dashed line in Fig. 3. Our experiments show that OU-PBVI's computed policies approach the best value of our hand-written, parameterized BSQ policy as N_p and $\|B\|$ increase.

The average runtimes for OU-PBVI ranged from 420s for 8 beliefs and 200 particles to 106, 800s for 1000 particles and 128 beliefs. At least 80% of the time was spent in proba-

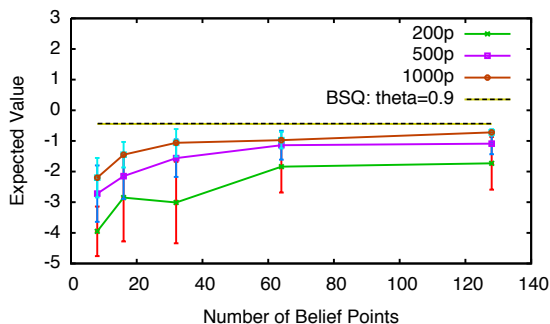


Figure 3: Expected values for OU-PBVI. Solid lines represent different numbers of particles used for the belief state representation (see text for details).

bilistic inference for carrying out belief propagation. Since we utilize PPLs to express OUPOMDPs, our approach will automatically scale with improvements in their inference engines. Indeed, using compiler techniques for improving inference in PPLs is an active area of research, and has shown speed-ups of up to 200x in preliminary experiments. We expect the runtimes of our algorithm to reduce to a fraction of the current estimates as a result of such advances.

These experiments demonstrate that our framework can be used to effectively express OUPOMDPs and solve them.

6 DISCUSSION

To the best of our knowledge, Moore (1985) presented the first comprehensive FOL formulation of actions that did not make the unique names assumption and allowed terms in the language to be partially observable, in a non-probabilistic framework. In Moore’s formulation actions could be executed by an agent only if they were “known” to it. This notion of epistemic feasibility of an action was also used in later work (Morgenstern, 1987; Davis, 1994, 2005). These approaches used a significantly larger axiomatization to address the problem of syntactically proving and communicating facts about knowledge. However, they cannot be used in open-universe probabilistic languages due to the requirement of reifying possible worlds and terms as objects in a universe. Further, they do not address the problems of expressing observability and action availability while conforming to a given agent specification.

Our formulation of action effects uses update rules similar to successor-state axioms (SSAs) (Reiter, 2001) with a significant enhancement: they allow compact expression of the so-called factored representations that are difficult to express using SSAs (Sanner and Kersting, 2010). Moreover, usually employed assumptions like having a “closed initial database” in that line of work preclude the possibility of expressing identity uncertainty: distinct terms like *Vesper* and *Fiancee(Bond)* can never represent the same object. Sanner and Kersting (2010) use this framework for first-order POMDPs and make the additional assumption

that all non-fluent terms are fully observable. They suggest a *same-as*(t_1, t_2) predicate for representing identity uncertainty between fluent terms. However, it is not clear how this predicate can be used in conjunction with their unique names axioms for actions, which assert that instances of an action applied on distinct terms must be distinct. The RDDL language (Sanner, 2010) used in recent probabilistic planning competitions can also express closed-universe POMDPs as relational extensions of DBNs, under the assumptions that a fixed set of predicates will be observed at every time step and that all ground terms are unique and known. Wang and Khardon (2010) present a relational representation for closed-universe POMDPs where action arguments have to be in a known 1-1 mapping with actual objects in the universe. Kaelbling and Lozano-Pérez (2013) present an approach where action specifications include preconditions in the form of belief-state fluents. However, the solution approach requires action-specific regression functions over the probabilities of such queries. The approaches discussed so far assume that a POMDP definition is available. Recent work by Doshi-Velez (2010) addresses the problem of learning the transition and observation distributions for an *unfactored* POMDP with a potentially unbounded number of states.

In contrast to these approaches, our formulation allows an agent to plan and act upon objects *discovered* through its sensors. We presented the first framework for accurately expressing OUPOMDPs and solving them. The central idea of our solution is that representing observations and decisions using terms with unique meanings clarifies communication without being restrictive. We utilized this idea to construct strategy trees reflecting the true capabilities of an agent—something that could not be achieved by extending the existing formalisms. We also showed that this framework facilitates general algorithms for solving a large class of decision problems that capture real-world situations and could not previously be expressed or solved. A number of directions exist for future work. The notion of high-level actions can be developed further. For instance, one could define an action that determines the maximum likelihood estimate for the position of the person who generated a reference, and takes a picture of that location. Such actions have to be specified outside the DTBLOG model since they need to execute queries on the model to construct their arguments. However, probabilistic effects of such actions have to be defined in the model in a manner consistent with their external definitions.

Acknowledgments

This research was supported in part by DARPA contracts W31P4Q-11-C-0083 and FA8750-14-C-0011, ONR grant N00014-12-1-0609 and ONR MURI award ONR-N00014-13-1-0341.

References

- Astrom, K. J. 1965. Optimal control of Markov decision processes with incomplete state estimation. *Journal of Mathematical Analysis and Applications* 10:174–205.
- Davis, E. 1994. Knowledge preconditions for plans. *J. Log. Comput.* 4(5):721–766.
- Davis, E. 2005. Knowledge and communication: A first-order theory. *AIJ* 166(1-2):81–139.
- Dean, T., and Kanazawa, K. 1989. A model for reasoning about persistence and causation. *Comput. Intell.* 5(3):142–150.
- Doshi-Velez, F. 2010. The infinite partially observable markov decision process. In *Neural Information Processing Systems (NIPS)*, volume 22.
- Goodman, N. D.; Mansinghka, V. K.; Roy, D. M.; Bonawitz, K.; and Tenenbaum, J. B. 2008. Church: A language for generative models. In *UAI-08*.
- Kaelbling, L. P., and Lozano-Pérez, T. 2013. Integrated task and motion planning in belief space. *I. J. Robotic Res.* 32(9-10):1194–1227.
- Kaelbling, L. P.; Littman, M. L.; and Cassandra, A. R. 1998. Planning and acting in partially observable stochastic domains. *AIJ* 101(1-2):99–134.
- Koller, D.; McAllester, D.; and Pfeffer, A. 1997. Effective Bayesian inference for stochastic programs. In *AAAI-97*.
- Levesque, H. J., and Lakemeyer, G. 2000. *The logic of knowledge bases*. MIT Press.
- Little, R. J. A., and Rubin, D. B. 2002. *Statistical analysis with missing data (second edition)*. Wiley.
- Milch, B.; Marthi, B.; Russell, S. J.; Sontag, D.; Ong, D. L.; and Kolobov, A. 2005. BLOG: Probabilistic models with unknown objects. In *Proc. IJCAI-05*, 1352–1359.
- Milch, B. C. 2006. *Probabilistic models with unknown objects*. Ph.D. Dissertation, UC Berkeley.
- Moore, R. C. 1985. *A Formal Theory of Knowledge and Action*. Ablex. 319–358.
- Morgenstern, L. 1987. Knowledge preconditions for actions and plans. In *Proc. IJCAI-87*, 867–874.
- Pineau, J.; Gordon, G.; Thrun, S.; et al. 2003. Point-based value iteration: An anytime algorithm for POMDPs. In *Proc. IJCAI-03*.
- Reiter, R. 2001. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. The MIT Press.
- Russell, S. J., and Norvig, P. 1995. *Artificial Intelligence - A Modern Approach*. Prentice Hall.
- Sanner, S., and Kersting, K. 2010. Symbolic dynamic programming for first-order POMDPs. In *Proc. AAAI-10*.
- Sanner, S. 2010. Relational dynamic influence diagram language (RDDL): Language description. http://users.cecs.anu.edu.au/~ssanner/IPPC_2011/RDDL.pdf.
- Srivastava, S.; Cheng, X.; Russell, S.; and Pfeffer, A. 2012. First-order open-universe POMDPs: Formulation and algorithms. Technical report, EECS-2013-243, EECS Department, UC Berkeley.
- Wang, C., and Khaldon, R. 2010. Relational partially observable MDPs. In *Proc. AAAI-10*.

A Hierarchical Switching Linear Dynamical System Applied to the Detection of Sepsis in Neonatal Condition Monitoring

Ioan Stanculescu
School of Informatics
University of Edinburgh
Edinburgh, UK, EH8 9AB

Christopher K.I. Williams
School of Informatics
University of Edinburgh
Edinburgh, UK, EH8 9AB

Yvonne Freer
Simpson Centre for Reproductive Health
Royal Infirmary of Edinburgh
Edinburgh, UK, EH16 4SA

Abstract

In this paper we develop a Hierarchical Switching Linear Dynamical System (HSLDS) for the detection of sepsis in neonates in an intensive care unit. The Factorial Switching LDS (FSLDS) of Quinn et al. (2009) is able to describe the observed vital signs data in terms of a number of discrete factors, which have either physiological or artifactual origin. In this paper we demonstrate that by adding a higher-level discrete variable with semantics sepsis/non-sepsis we can detect changes in the physiological factors that signal the presence of sepsis. We demonstrate that the performance of our model for the detection of sepsis is not statistically different from the auto-regressive HMM of Stanculescu et al. (2013), despite the fact that their model is given “ground truth” annotations of the physiological factors, while our HSLDS must infer them from the raw vital signs data.

1 INTRODUCTION

In condition monitoring, we are often interested in inferring when a dynamical system “switches” its mode of operation. Inside Neonatal Intensive Care Units (NICUs), one the most important “switches” is associated with the start of late onset neonatal sepsis (LONS). LONS is a bloodstream infection, usually bacterial, which generally occurs after the third day of life. It is a major cause of mortality, lifelong neurodisability and increased health care costs (Modi et al., 2009).

Since early clinical signs are subtle, making the diagnosis of infection is a great challenge. A deterioration of the baby’s condition prompts clinicians to take a

blood sample for laboratory testing. However, laboratory culture results can take up to a day before becoming available. This delay is known to prevent effective treatment (Griffin et al., 2003). Thus, a dependable early sepsis detector would have a major impact on NICU care. In this work, we discuss a solution which relies exclusively on vital signs monitoring data.

We propose a Hierarchical Switching Linear Dynamical System (HSLDS) to model a dynamical system with complex interactions between modes of operation. The structure of the model is shown in Fig. 1. In the HSLDS, the switch state is represented as a two-level discrete hierarchical structure. The top layer switch variables control the transition matrices used by the lower discrete layer, whose variables are assumed to be conditionally independent given the top layer. Conditioned on the hidden discrete structure, the model is a Linear Dynamical System (LDS), which models continuous hidden state variables and continuous observations. The observations are assumed to come from readings of the monitoring equipment.

The HSLDS can be applied for the real-world task of detecting neonatal sepsis. The discrete top layer determines the state of the infection and the lower-level discrete factors are baby-generated physiological events. The physiological events we monitor for sepsis detection are:

- *bradycardia*: a spontaneous fall in heart rate measurements (Figure 2a), and
- *desaturation*: a drop in the concentration of oxygen in arterial blood (Figure 2b).

The problem of detecting neonatal sepsis from monitoring data has been previously studied. Griffin et al. (2003) and Moorman et al. (2011) have found a positive skew in the inter-beat (RR) interval histograms in the hours before the clinical suspicion of sepsis, and an absence of skew during normal periods. They used this finding to build features subsequently fed to a logistic

regression classifier. However, this work does not use other vital signs apart from the heart rate and also assumes access to the high-frequency RR data. The work of Stanculescu et al. (2013) is probably closest to our approach. They propose using an auto-regressive HMM (AR-HMM) to capture trends of increased physiological event incidence. Unlike the HSLDS, their model uses annotations of physiological events as input, which limits the possibility of model deployment.

The main contributions of this work are: (i) to develop the FSLDS model of Quinn et al. (2009) into a HSLDS in a “deep learning” style by adding a set of higher-level variables to model correlations in the physiological factors in order to detect sepsis, and (ii) to demonstrate that the performance of our model for the detection of sepsis is almost as good as the auto-regressive HMM of Stanculescu et al. (2013), despite the fact that their model is given “ground truth” annotations of the physiological factors, while our HSLDS must infer them from the raw vital signs data.

The structure of the remainder of the paper is as follows: In Section 2 we describe the proposed model, and discuss inference, learning and related work. Section 3 explains how the HSLDS can be used to obtain early predictions about neonatal sepsis and inferences about clinical events. Experimental results are presented in Section 4 and we provide a discussion in Section 5.

2 THE HSLDS

In order to facilitate the introduction of our hierarchical model, we begin with a brief review of the Switching LDS (SLDS). The SLDS is a generative model for sequential data which switches between several different modes of operation. Each mode of operation is modelled as a LDS (Kalman filter), and thus the SLDS can be thought of as a dynamical mixture of LDS models. As the switch settings are hidden, often the main task is to recover them given the observations. Formally, at time t the SLDS has a discrete-continuous hybrid hidden state consisting of a hidden switch variable s_t and a hidden continuous state $\mathbf{x}_t \in \mathbb{R}^{d_x}$. This hybrid state attempts to explain how measurements $\mathbf{y}_t \in \mathbb{R}^{d_y}$ are generated. More precisely, the switch setting s_t determines the set of LDS parameters used at time t :

$$\mathbf{x}_t \sim \mathcal{N}(\mathbf{A}(s_t)\mathbf{x}_{t-1}, \mathbf{Q}(s_t)), \quad (1)$$

$$\mathbf{y}_t \sim \mathcal{N}(\mathbf{C}(s_t)\mathbf{x}_t, \mathbf{R}(s_t)), \quad (2)$$

where $\mathbf{A}(s_t)$ and $\mathbf{Q}(s_t)$ are the dynamics and dynamics noise covariance matrices, and $\mathbf{C}(s_t)$ and $\mathbf{R}(s_t)$ are the observation and observation noise covariance matrices. The switch settings are sampled from a Markov transition matrix $p(s_t|s_{t-1})$.

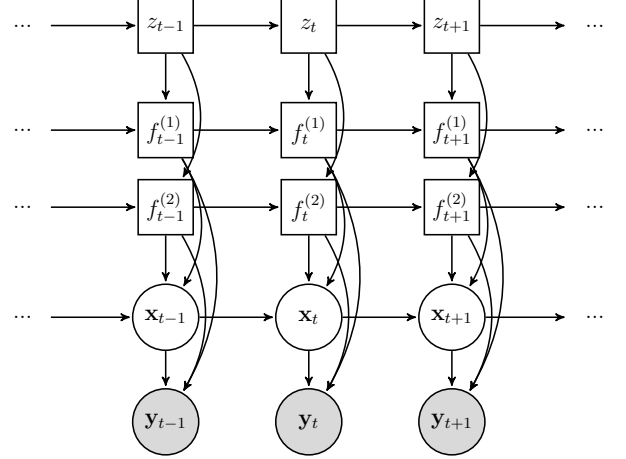


Figure 1: HSLDS with $K = 2$. Squares represent discrete variables and circles represent continuous ones. Shaded nodes are observed variables.

The FSLDS assumes a set of K discrete factors $f_t^{(1)}, f_t^{(2)}, \dots, f_t^{(K)}$ are collectively affecting the data. The model is obtained by representing the switch variable of the SLDS as the cross product $f_t^{(1)} \otimes f_t^{(2)} \otimes \dots \otimes f_t^{(K)}$. An important assumption made by the FSLDS is that the factors are a priori independent:

$$p(s_t|s_{t-1}) = \prod_{k=1}^K p(f_t^{(k)}|z_t, f_{t-1}^{(k)})$$

In the HSLDS, we propose relaxing this assumption by introducing a hierarchical structure for the discrete hidden variables. The discrete state is now represented by two layers of variables (see Figure 1). The top layer variable z_t controls the Markovian dynamics $p(f_t^{(\cdot)}|z_t, f_{t-1}^{(\cdot)})$ used by each factor. Conditional on the setting of the top layer switch variable z_t , the model becomes equivalent to an FSLDS. Thus, the HSLDS can be thought of as a dynamical mixture of FSLDS models. If we define a full expansion of the discrete hidden state as $s_t \triangleq z_t \otimes f_t^{(1)} \otimes f_t^{(2)} \otimes \dots \otimes f_t^{(K)}$, then the joint distribution of the HSLDS can be written as:

$$p(s_{1:T}, \mathbf{x}_{1:T}, \mathbf{y}_{1:T}) = p(s_1)p(\mathbf{x}_1|s_1)p(\mathbf{y}_1|\mathbf{x}_1, s_1) \prod_{t=2}^T p(s_t|s_{t-1})p(\mathbf{x}_t|\mathbf{x}_{t-1}, s_t)p(\mathbf{y}_t|\mathbf{x}_t, s_t), \quad (3)$$

where

$$p(s_1) = p(z_1) \prod_{k=1}^K p(f_1^{(k)}|z_1),$$

$$p(s_t|s_{t-1}) = p(z_t|z_{t-1}) \prod_{k=1}^K p(f_t^{(k)}|z_t, f_{t-1}^{(k)}),$$

$s_{1:T} \triangleq s_1, s_2, \dots, s_T$, and $\mathbf{x}_{1:T}$ and $\mathbf{y}_{1:T}$ are similarly defined.

Note that the top hidden layer is conditionally independent of the continuous variables given the factor settings: $\mathbf{x}_{1:T}, \mathbf{y}_{1:T} \perp\!\!\!\perp z_{1:T} | \mathbf{f}_{1:T}$, where we have defined $\mathbf{f}_t \triangleq [f_t^{(1)}, f_t^{(2)}, \dots, f_t^{(K)}]$. This simplifies both learning and inference.

2.1 RELATED WORK

The basic SLDS model has a long history, see e.g. Shumway and Stoffer (1991), and has been used in many applications. Factorization of the SLDS discrete state gives the Factorial Switching LDS (FSLDS), which has been used for neonatal condition monitoring by Williams et al. (2006) and Quinn et al. (2009), in speech recognition (Deng, 2006) and in music transcription (Cemgil et al., 2006). This mirrors the development of the factorial hidden Markov model (FHMM) of Ghahramani and Jordan (1997) from the standard HMM.

The HMM model has also been elaborated hierarchically by Fine and Singer (1998) to give the hierarchical hidden Markov model (HHMM). A similar construction can be used to create a hierarchical switching LDS (HSLDS). The only previous example of this model we are aware of in the literature is the work of Zoeter and Heskes (2003) which used a HSLDS for visualization of time-series data. Their motivation is to allow a successive refinement of a visualization, starting from projecting onto a single LDS with a two-dimensional (2-d) hidden space. This can be broken down into a SLDS of 2-d LDSes, and then each 2-d LDS can be further independently decomposed into a SLDS. Thus, a set of lower-level states correspond to one higher-level state. Also note that their use case involves interaction from the user to initialise the decomposition.

In contrast, we more naturally think of building our model bottom up, first identifying a set of factors for the FSLDS, and then modelling their correlations with a top-level variable. Notice that in our work the state of the top-level variable affects all of the second-level variables below it.

There are also some similarities between our work and the paper by Taylor et al. (2010). Under their approach, the \mathbf{x} dynamics are modelled by an Implicit Mixture of Conditional Restricted Boltzmann Machines (imCRBM). This is similar to us in that the CRBM part of the model uses a number of discrete latent variables (analogous to our \mathbf{f} 's) to affect the \mathbf{x} dynamics. The implicit mixture variable (analogous to our z) switches between different dynamics models. Of course, the details of the model are quite different as

it is in part undirected, and that there are no explicit discrete latent variable chains through time, instead these variables “hang off” the \mathbf{x} chain.

2.2 INFERENCE

Since real-time inference is the major concern in physiological condition monitoring, we are mainly interested in marginal filtering distributions. More precisely, we require sepsis predictions of the form $p(z_t | \mathbf{y}_{1:t})$ and clinical event posteriors $p(f_t^{(\cdot)} | \mathbf{y}_{1:t})$. These marginal posteriors can be immediately obtained from the filtering distribution of the fully expanded state $p(s_t | \mathbf{y}_{1:t})$. Thus, running SLDS inference suffices for HSLDS inference. Note that the more general goal of SLDS filtering is inferring $p(\mathbf{x}_{1:t}, s_{1:t} | \mathbf{y}_{1:t})$.

Exact SLDS inference requires computing Gaussian mixtures with a number of components exponential in the length of the sequence. Clearly, this is computationally intractable for most practical purposes (Lerner and Parr, 2001). Several approximate SLDS inference methods have been previously proposed: Gaussian sum approximations (Murphy, 1998; Barber and Mesot, 2007), Rao-Blackwellised Particle Filtering (Murphy and Russell, 2001; de Freitas et al., 2004), variational inference (Ghahramani and Hinton, 2000) or expectation propagation (Zoeter and Heskes, 2003).

Here, we apply the Gaussian Sum approximation described in Murphy (1998). The method ensures tractability by using moment matching to collapse a Gaussian mixture onto a single Gaussian. At any time step, each $p(\mathbf{x}_t | s_t, \mathbf{y}_{1:t})$ is approximated by a single Gaussian, which corresponds to $p(\mathbf{x}_t | \mathbf{y}_{1:t})$ being approximated by a mixture of Gaussians.

When the hidden discrete state is a cross-product of variables, we can speed up inference by allowing at most one variable to change its setting at each time step. This procedure has been previously discussed in Quinn et al. (2009) or Kolter and Jaakkola (2012).

A particular aspect of the baby monitoring application is the presence of several missing data sources. The treatment of this problem will be discussed in detail in Section 3.3.

2.3 LEARNING

HSLDS learning is similar to FSLDS learning to a large extent. Here, we first emphasize the most significant common aspects and then discuss HSLDS learning specifics.

For our application we assume that there are a number of interpretable regimes for which labelled data are available. Labelled data for the HSLDS model are of

the form $\{\mathbf{y}_t, z_t, f_t^{(1)}, f_t^{(2)}, \dots, f_t^{(K)}\}$.

As in the FSLDS case, the availability of labelled data makes learning equivalent to learning one LDS model for each switch setting. In general, we parameterise LDS dynamics as autoregressive processes and use Expectation Maximisation (EM) for training (Ghahramani and Hinton, 1996). Learning is performed independently for each factor, and then the fitted parameters are carefully combined for each switch setting. This procedure is greatly simplified by considering the interactions between factors. For instance, the activation of one factor might “overwrite” any effect of another factor on certain observation channels. In the neonatal monitoring application, domain knowledge is used to define a factor overwriting ordering, as further discussed in Section 3.2.

For the HSLDS in particular, we use the conditional independence between the continuous variables and the top layer discrete variables to further simplify learning. This means that the (parameters of the) continuous variable distributions (eqs. 1 and 2) do not depend on the setting of z_t .

A straightforward way of learning the Markov transition matrices for individual factors $p(f_t^{(\cdot)}|z_t, f_{t-1}^{(\cdot)})$ would be to make use of the labelled data and maximize the conditional likelihood $p(\mathbf{f}_{1:T}|z_{1:T})$. Estimates of the factor transition probabilities have the form:

$$p(f_t^{(\cdot)} = j|z_t = l, f_{t-1}^{(\cdot)} = i) = \frac{n_{ijl} + n_0}{\sum_{j'}(n_{ij'l} + n_0)}, \quad (4)$$

where n_{ijl} is the number of transitions from state i to state j for factor $f^{(\cdot)}$ under the z -regime l counted over all the training data. The constant count n_0 comes from placing a Dirichlet prior which prevents probabilities from being too close to zero.

However, we have found that an alternative “deep learning” style method can give rise to better results (Section 4.1). Although the \mathbf{f} data is available at training time, at test time these labels must be inferred from the \mathbf{y} data. Hence it makes sense to build a model which looks at the actual inferences of the factors, rather than the ground truth labels.

If \mathbf{Y} is the training set of sequences and the corresponding \mathbf{F} are treated as hidden variables, we could use EM and attempt to optimise $p(\mathbf{Y}|\mathbf{Z})$. The M-step is equivalent to maximizing the expected complete data log likelihood:

$$\mathcal{Q} = \mathbb{E}_{p(\mathbf{X}, \mathbf{F}|\mathbf{Y}, \mathbf{Z})} \log p(\mathbf{Y}, \mathbf{X}, \mathbf{F}|\mathbf{Z}), \quad (5)$$

where $p(\mathbf{X}, \mathbf{F}|\mathbf{Y}, \mathbf{Z})$ was computed in the preceding E-step using the old parameter settings. Factor transi-

tion estimates are of the form:

$$p(f_t^{(\cdot)} = j|z_t = l, f_{t-1}^{(\cdot)} = i) = \frac{\tilde{n}_{ijl} + n_0}{\sum_{j'}(\tilde{n}_{ij'l} + n_0)}, \quad (6)$$

where

$$\tilde{n}_{ijl} = \sum_t p(f_{t-1}^{(\cdot)} = i, f_t^{(\cdot)} = j|\mathbf{Y}, \mathbf{Z}) I(z_t = l)$$

is commonly referred to as a “soft” data count, I is the indicator function, and the sum is taken over all t ’s in the training data.

Running EM until convergence is likely to be unsatisfactory, as there are no guarantees that the learnt factor transition matrices would produce good factor posteriors. Our solution is to approximate $p(\mathbf{F}|\mathbf{Y}, \mathbf{Z})$, by $p_{FSLDS}(\mathbf{F}|\mathbf{Y})$. Here, the FSLDS model is trained using the standard learning routine of Quinn et al. (2009) and the factor models discussed in Section 3.2, and is thus unaware of the existence of multiple z -regimes. In practice, we found it sufficient to obtain “soft” counts of pairwise filtering marginals $p_{FSLDS}(f_{t-1}^{(\cdot)}, f_t^{(\cdot)}|\mathbf{y}_{1:t})$ for each training sequence. Since FSLDS posteriors do not depend on the learnt HSLDS parameters, the method is non-iterative.

This procedure follows ideas in the “deep learning” literature (Hinton et al., 2006) where layer-wise training of a model is carried out. Similar ideas can also be found e.g. in Karklin and Lewicki (2005) or Farhadi et al. (2009), although in all these cases the models are not for time series.

Finally, estimates of the Markov transition matrix $p(z_t|z_{t-1})$ are learnt from the z -labels. Also note that in the absence of the labelled data, unsupervised learning for the full model would be possible using EM.

3 AN HSLDS FOR NEONATAL CONDITION MONITORING

This section is concerned with applying the HSLDS for condition monitoring in NICUs. We begin with a brief description of baby monitoring, focusing on the early detection of neonatal sepsis. We then explain how the problem can be solved by formulating it as learning and inference in an HSLDS.

3.1 NICU MONITORING AND SEPSIS DETECTION

NICU babies are born several months prematurely and are intrinsically unstable. They are nursed in incubators, and their vital signs are continuously displayed on bedside screens. Clinicians apply their expertise to interpret patterns in the monitoring traces and use

this information in support of their diagnostic inference. The task is challenging for reasons including the amount, dimensionality and frequency of the data, and the need to analyse patient physiology across multiple time scales.

The present application focuses on the early detection of neonatal sepsis based on the information contained in the monitoring data. The hypothesis is that an increased incidence of baby generated physiological events is a symptom of sepsis. In current clinical practice, the laboratory result of a blood culture is taken as the “gold standard” for diagnosing neonatal sepsis. Here, we adopted the laboratory result interpretation proposed by Modi et al. (2009) and also discussed by Stanculescu et al. (2013).

The measurement channels used in this work monitor several vital physiological systems. The heart rate measures the cardiovascular system. It is available from two sources: the ECG leads - HR (beats per minute - bpm) and the pulse oximeter - PR (bpm). The core and peripheral temperatures, TC ($^{\circ}\text{C}$) and TP ($^{\circ}\text{C}$), monitor the thermoregulatory system. The saturation of oxygen in arterial blood, SO (%), reflects the evolution of the respiratory system. All channels are sampled second-by-second (1Hz).

Our data samples are monitoring windows with a duration of 30 hours, and fall into either a sepsis group or a control group. Sepsis samples have been chosen such that the time the positive blood sample was collected occurs precisely 24 hours after the start of the window. For control samples, there was no suspicion of sepsis in a consecutive 3 day period around the selected windows, and no blood sample had been taken for laboratory testing.

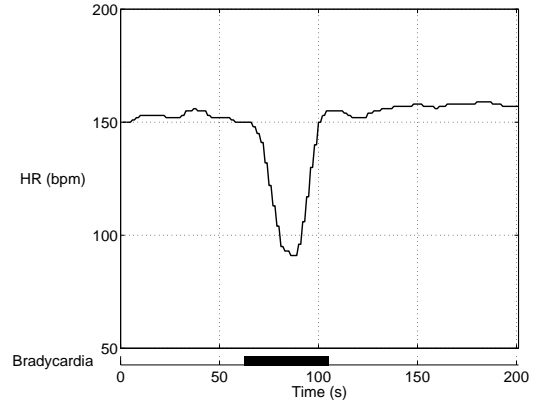
3.2 LEARNING A SEPSIS DETECTION MODEL

We now detail how the baby monitoring HSLDS is trained. We first discuss parameter fitting for the continuous variable distributions and then continue with learning the hidden discrete layers of the HSLDS.

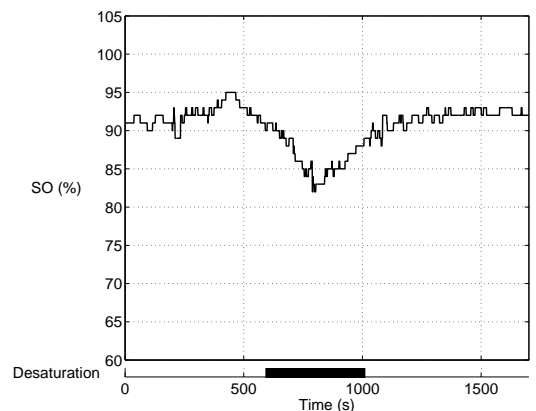
Learning continuous variable distributions

A natural classification of the regimes appearing in the NICU monitoring application is: *stability*, *known factors* and *unknown factors*.

Babies within the NICU are in a stable condition for much of the time, generally being asleep and motionless. We call this regime *stability* and separately fit univariate LDSes to each measurement channel. Thus, the dynamics parameters \mathbf{A} and \mathbf{Q} will have a block diagonal structure (see Quinn et al. (2009) for details).



(a) Bradycardia



(b) Desaturation

Figure 2: Examples of physiological events. They are notable for the lack of artifact.

When clinical events associated with stereotypical patterns occur on the monitoring traces, the regimes will be referred to as *known factors*. Here, we model two physiological events: bradycardias and desaturations (see Figure 2 for examples). Both are characterised by a drop in the monitored signal (a slowing of the heart rate for bradycardias, and a decrease in the saturation of oxygen in arterial blood for desaturations), after which measurements rise back. We model these factors as two-stage events. The first stage corresponds to measurements dropping and can be explained by an exponential decay, the discrete time equivalent of which is an $AR(1)$ process. To set the mean of the decay process we first compute the empirical distribution F of minimum channel measurements during events. The quantile q^* of F corresponding to $F(q^*) = 0.05$ is chosen to be the decay mean. In the second stage of the event the measurements rise back. This will be referred to as the recovery stage. Recovery dynamics are also modelled as an $AR(1)$ process, where the mean is now the same as the channel’s *stability* mean. The

Table 1: Overwriting Ordering of Factors

Channel	Bradycardia	Desaturation	X	Stability
HR	•		•	•
SO		•	•	•

parameters for both decay and recovery models are learnt by running EM, where we chose the dynamics initialisation $\mathbf{A} = \mathbf{0}$.

Finally, certain events cannot be explained by either stability or by any of the known factors. These patterns represent either novel dynamics or their low incidence makes them impractical to model as known factors. Here, we follow the approach of Quinn et al. (2009), where they propose a factor explaining these “known *unknowns*”, the X-factor. It shares the same dynamics matrix with stability, but uses an inflated system noise covariance matrix. As the X-factor can claim patterns of both physiology and artifact, we do not use it directly for inferring the presence of sepsis.

Once the factor models have been separately learnt, they are combined using the overwriting order shown in Table 1. For each measurement channel, factors placed towards the left of the table overwrite factors placed towards the right.

Learning discrete variable distributions

In the baby monitoring application the top discrete layer of the HSLDS models the state of the sepsis infection. Here, we assume z_t is a binary variable taking on values $z_t = \textit{sepsis}$ or $z_t = \textit{normal}$. We first explain how labels of the form $\{\mathbf{y}_t, z_t\}$ have been defined. We then discuss how these labels are used to train the HSLDS’s discrete variable layers.

The task of providing labels for the sepsis indicator variable is non-trivial. For patients in the sepsis group, clinicians only hold records for the exact time of the positive blood test. It is almost certain that the onset of the infection occurred in the hours prior to this time stamp. However, the onset cannot be assumed to be an instantaneous switch. The following labelling scheme has been proposed for samples belonging to the sepsis group; see Stanculescu et al. (2013). First, a period of 6 hours before the time of the positive blood test is labelled as *sepsis*. Second, we introduce a transition period during which the baby progresses from being in the *normal* state to being in the *sepsis* state. The transition period is defined as the 12 hour interval between 18 and 6 hours before the positive test. We do not assign a label for this period and it is not used for either training or testing the discrete layers of the HSLDS. Third, the monitoring data before the transition period (i.e. the first 6 hours of a sample in the

Table 2: Missing Data Sources Affecting Baby-generated Physiological Events.

	Bradycardia	Desaturation
Handling	•	•
Oximeter error		•
HR dropout	•	
SO dropout		•

sepsis group) is labelled as *normal*. Fourth, we do not assign a label to data after the positive test, as these measurements are likely to be affected by the patient’s response to treatment and have less relevance for the task of real-time sepsis detection. Finally, all the data in the control group is assigned the *normal* label.

Using the sepsis labels, an estimate of $p(z_t|z_{t-1})$ can be directly obtained using data counts. For learning the z -conditioned *known* factors’ transition matrices, we apply the procedure explained in Section 2.3; see eq. 5 and the surrounding text. The X-factor’s incidence is assumed to be independent of the state of the infection, and thus the factor transition matrix is copied from the previously learnt FSLDS.

3.3 INFERENCE WITH MISSING DATA

We reiterate that this work is centred on the idea of monitoring baby-generated bradycardias and desaturations in order to predict sepsis. However, there are periods of time during which labels for these events cannot be provided even by an expert annotator. We will treat such periods as missing data. There are three distinct sources of missing data: probe dropouts, oximeter errors and patient handling. We first describe these sources and then explain how inference can be performed during such periods.

During probe dropouts measurements are not available due to either malfunctioning or temporary removal of the monitoring devices. They can be readily recognised by the zero values on the recorded channels.

An oximeter error occurs when there is a disagreement between the HR and PR traces. This indicates a temporary unreliability of the SO trace, and thus the impossibility to monitor desaturations. Here, we adopt the approach in Stanculescu et al. (2013), where an automated oximeter error detection algorithm has been applied as a preprocessing step.

Patients are regularly handled by clinical staff (e.g. for changing nappies). During such episodes, we usually see an increased variability in the monitoring channels and often patterns of bradycardia or desaturation. We cannot distinguish whether such instances are caused merely by handling an extremely fragile baby, or they actually reflect the patient’s true

Table 3: Population Demographics: Gestation, Birth Weight (BW) and Post Partum Age

Group	Statistic	Gestation	BW	Age
Sepsis	mean	27.2 weeks	873 gr	14.5 days
	std.dev.	1.5 weeks	256 gr	8.5 days
Control	mean	26.7 weeks	837 gr	15.2 days
	std.dev.	1.7 weeks	139 gr	14 days

state of health. Thus, for sepsis detection we analyse only physiological events happening outside handling episodes. Our work still relies on having expert annotations for handling. Quinn et al. (2009) have shown that these episodes can be inferred by monitoring environmental channels such as the incubator’s humidity, but such channels have not been available in this work.

Table 2 shows how physiological events are affected by the presence of each missing data source.

For running inference with missing data, we extend the ideas in Quinn et al. (2009). Whenever a missing data source is present, the measurements do not carry information about the true physiology of the patient, and thus should not influence the hidden state estimates. The latter continue to evolve according to the dynamics equations, but without measurement update. Technically, rows of the observation matrix are set to zero whenever there is missing data on the corresponding measurement channel. For these channels the Kalman gain will be zero. Thus, the corresponding hidden continuous state dimensions will be estimated with increasing uncertainty before reaching the stable state of the Kalman filter.

4 EXPERIMENTS

This section describes the experiments we have performed to assess the neonatal condition monitoring model introduced in Section 3. The detection of sepsis is discussed in Section 4.1. Section 4.2 is concerned with the quality of physiological event posteriors.

The dataset we use in this work consists of data collected exclusively from very low birth weight patients (VLBW, birth weight < 1500 grams). It has been previously used by Stanculescu et al. (2013), and contains 36 monitoring samples equally split between the sepsis and the control groups. All sepsis samples come from different patients. In the control group we have two samples from each of 9 different babies. Three patients have samples in both groups, corresponding to a total of 24 different patients. The demographics of the two groups are shown in Table 3.

Expert annotations have been obtained for all the data. A summary of the annotation process is pro-

Table 4: Clinical Event Incidence

Event	Group	Incidence	Total	Median
Bradycardia	Sepsis	1718	24 hrs	39 sec
	Control	1133	12 hrs	35 sec
Desaturation	Sepsis	738	32 hrs	101 sec
	Control	231	11 hrs	132 sec
X-factor	Sepsis	226	10 hrs	94 sec
	Control	171	7 hrs	114 sec
Handling	Sepsis	204	44 hrs	530 sec
	Control	210	55 hrs	592 sec
Ox. err.	Sepsis	4051	45 hrs	16 sec
	Control	3395	36 hrs	18 sec

vided in Table 4. The total amount of data for each group is $18 \times 30 = 540$ hours and only baby generated physiological events have been considered. Importantly, the incidence of baby generated bradycardias and desaturations is higher in the sepsis group. As expected, the differences for the X-factor are much smaller. In terms of missing data sources, the amounts of handling and oximeter error are similar between patient groups. Probe dropout statistics are different for each channel, but on average we lack observations for 2% of the time. In addition, a *stability* period of 15–30 minutes was marked near the start of each sample.

In order to reduce bias, we test our predictions using N -fold cross-validation. Considering the size of our dataset we decided to use $N = 9$ folds. Each fold contains 4 data samples, 2 from each patient group. The 2 control samples are chosen such that they belong to the same patient. Apart from these constraints, the folds have been randomly chosen.

4.1 SEPSIS DETECTION

To gain a better understanding of the HSLDS’s effectiveness, we compare its predictions against filtering results obtained with the AR-HMM model of Stanculescu et al. (2013). While the HSLDS infers the posterior distributions of bradycardias and desaturations, the AR-HMM uses expert annotations of these events as input. Note that in the AR-HMM it was possible to run inference exactly and also marginalise over the missing data exactly. For the purposes of this work, the central question is how well the HSLDS inferences match the AR-HMM ones.

In the following we discuss two HSLDS models. The HSLDS learnt as explained in Section 2.3 will be referred to as HSLDSdeep. We will compare it against an HSLDS where the factor transitions for baby-generated events are learnt directly from the expert annotations, HSLDSkf (known factors).

We provide the second-by-second sepsis inferences produced by both the AR-HMM and HSLDSdeep in Fig-

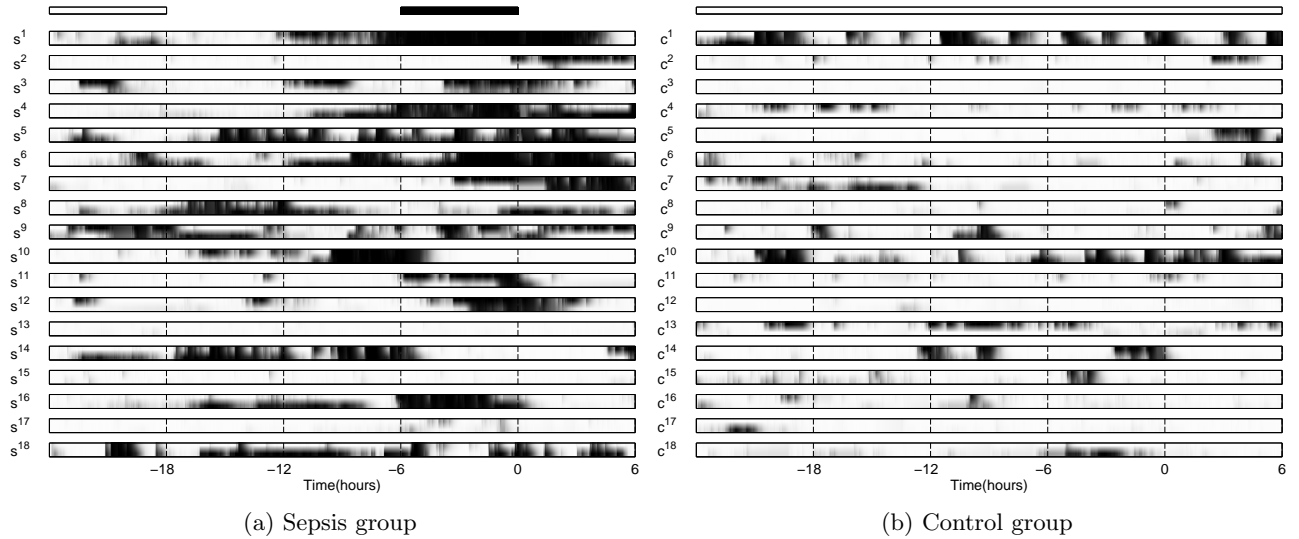


Figure 3: Sepsis filtering distributions obtained using 9-fold Cross-Validation. On the x -axis, 0 denotes the time the positive blood sample was taken. For each group, the top row represents the sepsis labelling: normal periods are white (probability 0), sepsis periods are black (probability 1); transitioning and treatment periods are not assigned labels. For each data sample the top row corresponds to the AR-HMM model, the bottom row corresponds to HSLDSdeep.

Table 5: Sepsis Inference Summaries Using 9-fold Cross-Validation

Model	Second-by-second		Episode-based	
	AUC	EER	AP	F-score
AR-HMM	0.72	0.34	0.62	0.65
HSLDSdeep	0.69	0.37	0.51	0.54
HSLDSkf	0.62	0.41	0.45	0.47

Figure 3. In general, there is strong correlation between the predictions of the two models and we find the inferences of HSLDSdeep to be a good match to the AR-HMM ones. However, in samples s^2 , s^7 and s^{11} HSLDSdeep detects sepsis noticeably later than the AR-HMM, and in samples s^4 and s^6 it does so earlier. In the control group, HSLDSdeep does slightly worse on samples c^7 and c^{18} , but outperforms the AR-HMM on samples c^4 and c^{13} .

For quantifying those results, we project the inferences onto two different metrics. This opens the possibility to reveal different aspects of performance.

Firstly, we are mainly interested in the *second-by-second* inferences produced by our hierarchical models and use the z -labels to draw ROC curves. The AUC (area under the ROC curve) and EER¹ computed by aggregating predictions over folds are shown in Ta-

¹EER is the error rate computed for the threshold at which the false positive rate (FPR) equals the false negative rate (FNR).

ble 5. Compared to HSLDSkf, HSLDSdeep produced results much closer to the AR-HMM benchmark.

We obtained more insight into how the HSLDS predictions compare against the AR-HMM results via an N -fold cross-validated paired t test on the AUC. We found the performance difference between the AR-HMM and our proposed HSLDSdeep model not to be statistically significant ($p = 0.552$). This is a good indication that the HSLDSdeep model can be used instead of the AR-HMM, and thus significantly reduce the need for expert input needed to detect sepsis. At the same time the performance difference between the AR-HMM and the HSLDSkf model is statistically significant ($p = 0.0064$). This suggests HSLDSkf should not be used instead of the AR-HMM.

Secondly, we analyse the inferred *episodes* of infection and draw precision-recall (PR) curves. This analysis has been proposed by Stanculescu et al. (2013), where they argue that it could be more relevant in clinical practice than a second-by-second one. Here we report average precision (AP) and the maximum F-score (see Table 5). Again, the performance of HSLDSdeep is closer to the AR-HMM than the HSLDSkf.

4.2 PHYSIOLOGICAL EVENT POSTERIORS

We can obtain filtering distributions for physiological events by marginalising the sepsis variable from

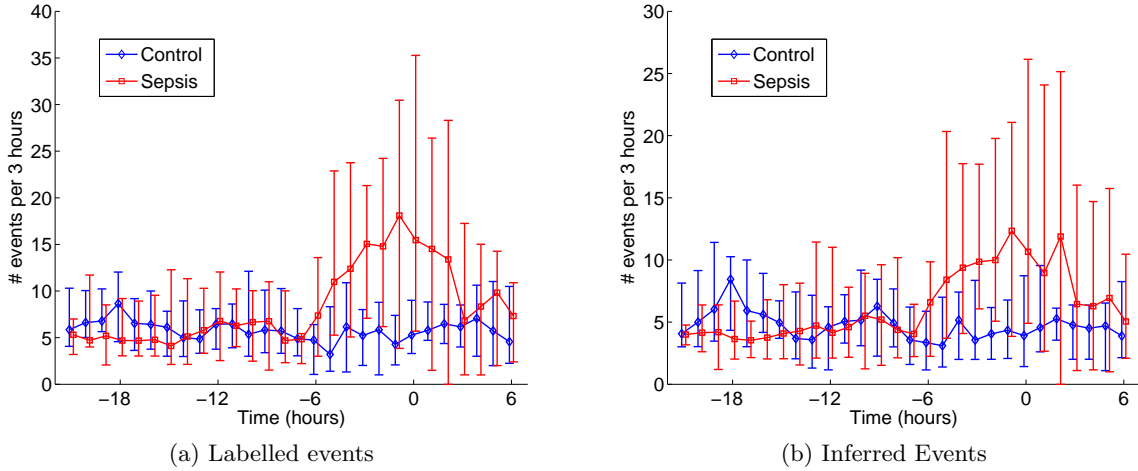


Figure 4: Median weighted number of true and inferred bradycardias separately computed for each patient group. The counts were computed hourly and summarize the preceding 3 hour period. Error bars mark first and third quartiles. The small offset between the two patient groups was used to improve readability.

Table 6: Factor Inference Summaries Using 9-fold Cross-Validation

		Brady.	Desat.	X
FSLDS	AUC	0.85	0.81	0.63
	EER	0.21	0.28	0.40
HSLDSdeep	AUC	0.86	0.82	0.60
	EER	0.21	0.27	0.42

HSLDS posteriors. As we have labelled data for the predicted factors, we can compare HSLDS posteriors against FSLDS ones. Summary results computed by aggregating predictions obtained with 9-fold cross-validation are shown in Table 6. Even though the FSLDS has been trained solely for inferring clinical events, there is very little difference between the performance of the two models.

Bradycardia and X-factor inferences obtained using an FSLDS have been previously assessed in (Quinn et al., 2009). The bradycardia results reported here are very similar to that work, but X-factor predictions are worse. Results on oxygen desaturation have not been previously reported.

We also found it interesting to compare the true incidence of baby-generated physiological events against the inferred one. For this purpose we obtained inferred events by binarising factor posteriors. Figure 4 shows a comparative visualisation of the time evolution of annotated and inferred bradycardias. The counts have been weighted in accordance to the amount of missing data in the analysed 3 hour periods. On both plots, there is a clear increase in the incidence of bradycardias in the hours before the sepsis diagnosis.

5 CONCLUSION

In this paper, we have proposed a framework for condition monitoring in situations when the factors that govern the data can be organised in a hierarchy. The structure of our model allows domain knowledge to be naturally incorporated. In addition, we have described a “deep learning” inspired training method.

The effectiveness of our model has been demonstrated for the difficult task of detecting the onset of sepsis in NICU patients. When compared against an AR-HMM model which heavily relies on expert annotations, we found the performance difference not to be statistically significant.

There are several directions in which this work could be extended. It would be interesting to run (H)SLDS smoothing, e.g. as described by Barber and Mesot (2007). This would prove useful both as a retrospective analysis of sepsis detection, and for refining our approach to learning factor transitions. Explicit modelling of event duration could improve the results, as demonstrated by Stanculescu et al. (2013). While we showed that the HSLDS performs similarly to the AR-HMM, sepsis predictions still need improvement. Finally, the X-factor predictions indicate more work could be done on novelty detection.

Acknowledgements

We would like to thank Prof. Neil McIntosh for providing expert insight and supervising with data annotation. Author IS was funded by the Scottish Informatics and Computer Science Alliance and the Engineering and Physical Sciences Research Council.

References

- Barber, D. and Mesot, B. (2007). A Novel Gaussian Sum Smoother for Approximate Inference in Switching Linear Dynamical Systems. In Schölkopf, B., Platt, J., and Hoffman, T., editors, *Advances in Neural Information Processing Systems 19*, pages 89–96. MIT Press, Cambridge, MA.
- Cemgil, A. T., Kappen, H. J., and Barber, D. (2006). A generative model for music transcription. *IEEE Transactions on Audio, Speech and Language Processing*, 14(2):679–694.
- de Freitas, N., Dearden, R., Hutter, F., Morales-Menendez, R., Mutch, J., and Poole, D. (2004). Diagnosis by a waiter and a Mars explorer. *Proceedings of the IEEE*, 92(3):455–468.
- Deng, L. (2006). *Dynamic Speech Models: Theory, Algorithms, and Applications*. Synthesis Lectures on Speech and Audio Processing. Morgan & Claypool Publishers.
- Farhadi, A., Endres, I., Hoiem, D., and Forsyth, D. (2009). Describing objects by their attributes. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Fine, S. and Singer, Y. (1998). The Hierarchical Hidden Markov Model: Analysis and Applications. In *Machine Learning*, pages 41–62.
- Ghahramani, Z. and Hinton, G. E. (1996). Parameter Estimation for Linear Dynamical Systems. Technical report, University of Toronto.
- Ghahramani, Z. and Hinton, G. E. (2000). Variational Learning for Switching State-Space Models. *Neural Computation*, 12(4):831–864.
- Ghahramani, Z. and Jordan, M. I. (1997). Factorial Hidden Markov Models. *Machine Learning*, 29:245–273.
- Griffin, M. P., O’Shea, T. M., Bissonette, E. A., Harrell, F. E., Lake, D. E., and Moorman, J. R. (2003). Abnormal Heart Rate Characteristics Preceding Neonatal Sepsis and Sepsis-Like Illness. *Pediatric Res*, 53(6):920–6.
- Hinton, G. E., Osindero, S., and Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554.
- Karklin, Y. and Lewicki, M. S. (2005). A hierarchical Bayesian model for learning non-linear statistical regularities in non-stationary natural signals. *Neural Computation*, 17(2):397–423.
- Kolter, J. Z. and Jaakkola, T. (2012). Approximate Inference in Additive Factorial HMMs with Application to Energy Disaggregation. In Lawrence, N. D. and Girolami, M., editors, *AISTATS*, volume 22 of *JMLR Proceedings*, pages 1472–1482. JMLR.org.
- Lerner, U. and Parr, R. (2001). Inference in hybrid networks: Theoretical limits and practical algorithms. In *UAI*, pages 310–318.
- Modi, N., Doré, C. J., Saraswatula, A., Richards, M., Bamford, K. B., Coello, R., and Holmes, A. (2009). A case definition for national and international neonatal bloodstream infection surveillance. *Archives of Disease in Childhood - Fetal and Neonatal Edition*, 94(1):F8–F12.
- Moorman, J. R., Carlo, W. A., Kattwinkel, J., Schelonka, R. L., Porcelli, P. J., Navarrete, C. T., Bancalari, E., Aschner, J. L., Walker, M. W., Perez, J. A., Palmer, C., Stukenborg, G. J., Lake, D. E., and O’Shea, T. M. (2011). Mortality Reduction by Heart Rate Characteristic Monitoring in Very Low Birth Weight Neonates: A Randomized Trial. *The Journal of Pediatrics*, 159(6):900 – 906.e1.
- Murphy, K. and Russell, S. (2001). Rao-Blackwellised Particle Filtering for Dynamic Bayesian Networks. In A. Doucet, N. d. F. and Gordon, N., editors, *Sequential Monte Carlo in Practice*. Springer-Verlag.
- Murphy, K. P. (1998). Switching Kalman Filters. Technical report, U.C. Berkeley.
- Quinn, J. A., Williams, C. K. I., and McIntosh, N. (2009). Factorial Switching Linear Dynamical Systems Applied to Physiological Condition Monitoring. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(9):1537–1551.
- Shumway, R. and Stoffer, D. (1991). Dynamic linear models with switching. *J. of the American Statistical Association*, 86:763–769.
- Stanculescu, I., Williams, C. K. I., and Freer, Y. (2013). Autoregressive Hidden Markov Models for the Early Detection of Neonatal Sepsis. *Biomedical and Health Informatics, IEEE Journal of*. DOI 10.1109/JBHI.2013.2294692.
- Taylor, G. W., Sigal, L., Fleet, D., and Hinton, G. E. (2010). Dynamic binary latent variable models for 3D human pose tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2010*.
- Williams, C. K. I., Quinn, J. A., and McIntosh, N. (2006). Factorial Switching Kalman Filters for Condition Monitoring in Neonatal Intensive Care. In Weiss, Y., Schölkopf, B., and Platt, J., editors, *Advances in Neural Information Processing Systems 18*. MIT Press.
- Zoeter, O. and Heskes, T. (2003). Hierarchical visualization of time-series data using switching linear dynamical systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:1201–1214.

A Spectral Algorithm for Learning Class-Based n -gram Models of Natural Language

Karl Stratos[†]

Do-kyum Kim[‡]

Michael Collins[†]

Daniel Hsu[†]

[†]Department of Computer Science, Columbia University, New York, NY 10027

[‡]Department of Computer Science and Engineering, University of California–San Diego, La Jolla, CA 92093

Abstract

The Brown clustering algorithm (Brown *et al.*, 1992) is widely used in natural language processing (NLP) to derive lexical representations that are then used to improve performance on various NLP problems. The algorithm assumes an underlying model that is essentially an HMM, with the restriction that each word in the vocabulary is emitted from a single state. A greedy, bottom-up method is then used to find the clustering; this method does not have a guarantee of finding the correct underlying clustering. In this paper we describe a new algorithm for clustering under the Brown *et al.* model. The method relies on two steps: first, the use of canonical correlation analysis to derive a low-dimensional representation of words; second, a bottom-up hierarchical clustering over these representations. We show that given a sufficient number of training examples sampled from the Brown *et al.* model, the method is guaranteed to recover the correct clustering. Experiments show that the method recovers clusters of comparable quality to the algorithm of Brown *et al.* (1992), but is an order of magnitude more efficient.

1 INTRODUCTION

There has recently been great interest in the natural language processing (NLP) community in methods that derive lexical representations from large quantities of unlabeled data (Brown *et al.*, 1992; Pereira *et al.*, 1993; Ando and Zhang, 2005; Liang, 2005; Turian *et al.*, 2010; Dhillon *et al.*, 2011; Collobert *et al.*, 2011; Mikolov *et al.*, 2013a,b). These representations can be used to improve accuracy on various NLP problems, or to give significant reductions in the number of training examples required for learning. The Brown clustering algorithm (Brown *et al.*, 1992) is one of the most widely used algorithms for this task. Brown clustering representations have been shown to be useful in a

diverse set of problems including named-entity recognition (Miller *et al.*, 2004; Turian *et al.*, 2010), syntactic chunking (Turian *et al.*, 2010), parsing (Koo *et al.*, 2008), and language modeling (Kneser and Ney, 1993; Gao *et al.*, 2001).

The Brown clustering algorithm assumes a model that is essentially a hidden Markov model (HMM), with a restriction that each word in the vocabulary can only be emitted from a single state in the HMM (i.e., there is a deterministic mapping from words to underlying states). The algorithm uses a greedy, bottom-up method in deriving the clustering. This method is a heuristic, in that there is no guarantee of recovering the correct clustering. In practice, the algorithm is quite computationally expensive: for example in our experiments, the implementation of Liang (2005) takes over 22 hours to derive a clustering from a dataset with 205 million tokens and 300,000 distinct word types.

This paper introduces a new algorithm for clustering under the Brown *et al.* model (henceforth, the Brown model). Crucially, under an assumption that the data is generated from the Brown model, our algorithm is guaranteed to recover the correct clustering when given a sufficient number of training examples (see the theorems in Section 5). The algorithm draws on ideas from canonical correlation analysis (CCA) and agglomerative clustering, and has the following simple form:

1. Estimate a normalized covariance matrix from a corpus and use singular value decomposition (SVD) to derive low-dimensional vector representations for word types (Figure 4).
2. Perform a bottom-up hierarchical clustering of these vectors (Figure 5).

In our experiments, we find that our clusters are comparable to the Brown clusters in improving the performance of a supervised learner, but our method is significantly faster. For example, both our clusters and Brown clusters improve the F1 score in named-entity recognition (NER) by 2-3 points, but the runtime of our method is around 10 times faster than the Brown algorithm (Table 3).

The paper is structured as follows. In Section 2, we discuss

Input: corpus with N tokens of n distinct word types $w^{(1)}, \dots, w^{(n)}$ ordered by decreasing frequency; number of clusters m .

Output: hierarchical clustering of $w^{(1)}, \dots, w^{(n)}$.

1. Initialize active clusters $\mathcal{C} = \{\{w^{(1)}\}, \dots, \{w^{(m)}\}\}$.
2. For $i = m + 1$ to $n + m - 1$:
 - (a) If $i \leq n$: set $\mathcal{C} = \mathcal{C} \cup \{\{w^{(i)}\}\}$.
 - (b) Merge $c, c' \in \mathcal{C}$ that cause the smallest decrease in the likelihood of the corpus.

Figure 1: A standard implementation of the Brown clustering algorithm.

related work. In Section 3, we establish the notation we use throughout. In Section 4, we define the Brown model. In Section 5, we present the main result and describe the algorithm. In Section 6, we report experimental results.

2 BACKGROUND

2.1 THE BROWN CLUSTERING ALGORITHM

The Brown clustering algorithm (Brown *et al.*, 1992) has been used in many NLP applications (Koo *et al.*, 2008; Miller *et al.*, 2004; Liang, 2005). We briefly describe the algorithm below; a part of the description was taken from Koo *et al.* (2008).

The input to the algorithm is a corpus of text with N tokens of n distinct word types. The algorithm initializes each word type as a distinct cluster, and repeatedly merges the pair of clusters that cause the smallest decrease in the likelihood of the corpus according to a discrete hidden Markov model (HMM). The observation parameters of this HMM are assumed to satisfy a certain disjointedness condition (Assumption 4.1). We will explicitly define the model in Section 4.

At the end of the algorithm, one obtains a hierarchy of word types which can be represented as a binary tree as in Figure 2. Within this tree, each word is uniquely identified by its path from the root, and this path can be compactly represented with a bit string. In order to obtain a clustering of the words, we select all nodes at a certain depth from the root of the hierarchy. For example, in Figure 2 we might select the four nodes at depth 2 from the root, yielding the clusters $\{\text{apple}, \text{pear}\}$, $\{\text{Apple}, \text{IBM}\}$, $\{\text{bought}, \text{run}\}$, and $\{\text{of}, \text{in}\}$. Note that the same clustering can be obtained by truncating each word’s bit string to a 2-bit prefix. By using prefixes of various lengths, we can produce clusterings of different granularities.

A naive implementation of this algorithm has runtime $O(n^5)$. Brown *et al.* (1992) propose a technique to reduce the runtime to $O(n^3)$. Since this is still not acceptable

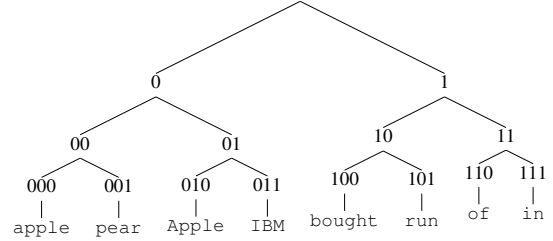


Figure 2: An example of a Brown word-cluster hierarchy taken from Koo *et al.* (2008). Each node in the tree is labeled with a bit string indicating the path from the root node to that node, where 0 indicates a left branch and 1 indicates a right branch.

for large values of n , a common trick used for practical implementation is to specify the number of active clusters $m \ll n$, for example, $m = 1000$. A sketch of this implementation is shown in Figure 1. Using this technique, it is possible to achieve $O(N + nm^2)$ runtime. We note that our algorithm in Figure 5 has a similar form and asymptotic runtime, but is empirically much faster. We discuss this issue in Section 6.3.1.

In this paper, we present a very different algorithm for deriving a word hierarchy based on the Brown model. In all our experiments, we compared our method against the highly optimized implementation of the Brown algorithm in Figure 1 by Liang (2005).

2.2 CCA AND AGGLOMERATIVE CLUSTERING

Our algorithm in Figure 4 operates in a fashion similar to the mechanics of CCA. CCA is a statistical technique used to maximize the correlation between a pair of random variables (Hotelling, 1936). A central operation in CCA to achieve this maximization is SVD; in this work, we also critically rely on SVD to recover the desired parameters.

Recently, it has been shown that one can use CCA-style algorithms, so-called spectral methods, to learn HMMs in polynomial sample/time complexity (Hsu *et al.*, 2012). These methods will be important to our goal since the Brown model can be viewed as a special case of an HMM.

We briefly note that one can view our approach from the perspective of spectral clustering (Ng *et al.*, 2002). A spectral clustering algorithm typically proceeds by constructing a graph Laplacian matrix from the data and performing a standard clustering algorithm (e.g., k -means) on reduced-dimensional points that correspond to the top eigenvalues of the Laplacian. We do not make use of a graph Laplacian, but we do make use of spectral methods for dimensionality reduction before clustering.

Agglomerative clustering refers to hierarchical grouping of n points using a bottom-up style algorithm (Ward Jr, 1963; Shanbehzadeh and Ogunbona, 1997). It is commonly used for its simplicity, but a naive implementation

requires $O(dn^3)$ time where d is the dimension of a point. Franti *et al.* (2000) presented a faster algorithm that requires $O(\gamma dn^2)$ time where γ is a data-dependent quantity which is typically much smaller than n . In our work, we use a variant of this last approach that has runtime $O(\gamma dmn)$ where $m \ll n$ is the number of active clusters we specify (Figure 5). We also remark that under our derivation, the dimension d is always equal to m , thus we express the runtime simply as $O(\gamma nm^2)$.

3 NOTATION

Let $[n]$ denote the set $\{1, \dots, n\}$. Let $[[\Gamma]]$ denote the indicator of a predicate Γ , taking value 1 if Γ is true and 0 otherwise. Given a matrix M , we let \sqrt{M} denote its element-wise square-root and M^+ denote its Moore-Penrose pseudoinverse. Let $I_{m \times m} \in \mathbb{R}^{m \times m}$ denote the identity matrix. Let $\text{diag}(v)$ denote the diagonal matrix with the vector $v \in \mathbb{R}^m$ appearing on its diagonal. Finally, let $\|v\|$ denote the Euclidean norm of a vector v , and $\|M\|$ denote the spectral norm of a matrix M .

4 BROWN MODEL DEFINITION

A Brown model is a 5-tuple (n, m, π, t, o) for integers n, m and functions π, t, o where

- $[n]$ is a set of states that represent word types.
- $[m]$ is a set of states that represent clusters.
- $\pi(c)$ is the probability of generating $c \in [m]$ in the first position of a sequence.
- $t(c'|c)$ is the probability of generating $c' \in [m]$ given $c \in [m]$.
- $o(x|c)$ is the probability of generating $x \in [n]$ given $c \in [m]$.

In addition, the model makes the following assumption on the parameters $o(x|c)$. This assumption comes from Brown *et al.* (1992) who require that the word clusters partition the vocabulary.

Assumption 4.1 (Brown *et al.* assumption). *For each $x \in [n]$, there is a unique $C(x) \in [m]$ such that $o(x|C(x)) > 0$ and $o(x|c) = 0$ for all $c \neq C(x)$.*

In other words, the model is a discrete HMM with a many-to-one deterministic mapping $C : [n] \rightarrow [m]$ from word types to clusters. Under the model, a sequence of N tokens $(x_1, \dots, x_N) \in [n]^N$ has probability

$$p(x_1, \dots, x_N) = \pi(C(x_1)) \times \prod_{i=1}^N o(x_i|C(x_i)) \times \prod_{i=1}^{N-1} t(C(x_{i+1})|C(x_i))$$

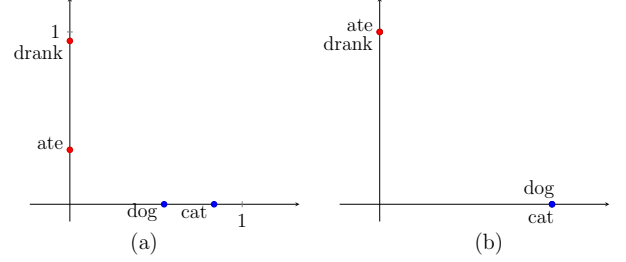


Figure 3: Illustration of our clustering scheme. (a) Original rows of \sqrt{O} . (b) After row-normalization.

An equivalent definition of a Brown model is given by organizing the parameters in matrix form. Under this definition, a Brown model has parameters (π, T, O) where $\pi \in \mathbb{R}^m$ is a vector and $T \in \mathbb{R}^{m \times m}, O \in \mathbb{R}^{n \times m}$ are matrices whose entries are set to:

- $\pi_c = \pi(c)$ for $c \in [m]$
- $T_{c',c} = t(c'|c)$ for $c, c' \in [m]$
- $O_{x,c} = o(x|c)$ for $c \in [m], x \in [n]$

Throughout the paper, we will assume that T, O have rank m . The following is an equivalent reformulation of Assumption 4.1 and will be important to the derivation of our algorithm.

Assumption 4.2 (Brown *et al.* assumption). *Each row of O has exactly one non-zero entry.*

5 CLUSTERING UNDER THE BROWN MODEL

In this section, we develop a method for clustering words based on the Brown model. The resulting algorithm is a simple two-step procedure: an application of SVD followed by agglomerative hierarchical clustering in Euclidean space.

5.1 AN OVERVIEW OF THE APPROACH

Suppose the parameter matrix O is known. Under Assumption 4.2, a simple way to recover the correct word clustering is as follows:

1. Compute $\bar{M} \in \mathbb{R}^{n \times m}$ whose rows are the rows of \sqrt{O} normalized to have length 1.
2. Put words x, x' in the same cluster iff $\bar{M}_x = \bar{M}_{x'}$, where \bar{M}_x is the x -th row of \bar{M} .

This works because Assumption 4.2 implies that the rows of \sqrt{O} corresponding to words from the same cluster lie along the same coordinate-axis in \mathbb{R}^m . Row-normalization puts these rows precisely at the standard basis vectors. See Figure 3 for illustration.

In Section 5.2, we prove that the rows of \sqrt{O} can be recovered, up to an orthogonal transformation $Q \in \mathbb{R}^{m \times m}$, just from unigram and bigram word probabilities (which can be estimated from observed sequences). It is clear that the correctness of the above procedure is unaffected by the orthogonal transformation. Let M denote the row-normalized form of $\sqrt{O}Q^\top$: then M still satisfies the property that $M_x = M_{x'}$ iff x, x' belong to the same cluster. We give an algorithm to estimate this M from a sequence of words in Figure 4.

5.2 SPECTRAL ESTIMATION OF OBSERVATION PARAMETERS

To derive a method for estimating the observation parameter \sqrt{O} (up to an orthogonal transformation), we first define the following random variables to model a single random sentence. Let $(X_1, \dots, X_N) \in [n]^N$ be a random sequence of tokens drawn from the Brown model, along with the corresponding (hidden) cluster sequence $(C_1, \dots, C_N) \in [m]^N$; independently, pick a position $I \in [N-1]$ uniformly at random. Let $B \in \mathbb{R}^{n \times n}$ be a matrix of bigram probabilities, $u, v \in \mathbb{R}^n$ vectors of unigram probabilities, and $\tilde{\pi} \in \mathbb{R}^m$ a vector of cluster probabilities:

$$\begin{aligned} B_{x,x'} &:= P(X_I = x, X_{I+1} = x') & \forall x, x' \in [n] \\ u_x &:= P(X_I = x) & \forall x \in [n] \\ v_x &:= P(X_{I+1} = x) & \forall x \in [n] \\ \tilde{\pi}_c &:= P(C_I = c) & \forall c \in [m]. \end{aligned}$$

We assume that $\text{diag}(\tilde{\pi})$ has rank m ; note that this assumption is weaker than requiring $\text{diag}(\pi)$ to have rank m . We will consider a matrix $\Omega \in \mathbb{R}^{n \times n}$ defined as

$$\Omega := \text{diag}(u)^{-1/2} B \text{diag}(v)^{-1/2} \quad (1)$$

Theorem 5.1. *Let $U \in \mathbb{R}^{n \times m}$ be the matrix of m left singular vectors of Ω corresponding to nonzero singular values. Then there exists an orthogonal matrix $Q \in \mathbb{R}^{m \times m}$ such that $U = \sqrt{O}Q^\top$.*

To prove Theorem 5.1, we need to examine the structure of the matrix Ω . The following matrices $A, \tilde{A} \in \mathbb{R}^{n \times m}$ will be important for this purpose:

$$\begin{aligned} A &= \text{diag}(O\tilde{\pi})^{-1/2} O \text{diag}(\tilde{\pi})^{1/2} \\ \tilde{A} &= \text{diag}(OT\tilde{\pi})^{-1/2} OT \text{diag}(\tilde{\pi})^{1/2} \end{aligned}$$

The first lemma shows that Ω can be decomposed into A and \tilde{A}^\top .

Lemma 5.1. $\Omega = A\tilde{A}^\top$.

Proof. It can be algebraically verified from the definition of B, u, v that $B = O \text{diag}(\tilde{\pi})(OT)^\top$, $u = O\tilde{\pi}$, and $v = OT\tilde{\pi}$. Plugging in these expressions in Eq. (1), we have

$$\begin{aligned} \Omega &= \text{diag}(O\tilde{\pi})^{-1/2} O \text{diag}(\tilde{\pi})^{1/2} \\ &\quad (\text{diag}(OT\tilde{\pi})^{-1/2} OT \text{diag}(\tilde{\pi})^{1/2})^\top = A\tilde{A}^\top. \quad \square \end{aligned}$$

The second lemma shows that A is in fact the desired matrix. The proof of this lemma crucially depends on the disjoint-cluster assumption of the Brown model.

Lemma 5.2. $A = \sqrt{O}$ and $A^\top A = I_{m \times m}$.

Proof. By Assumption 4.2, the x -th entry of $O\tilde{\pi}$ has value $O_{x,C(x)} \times \tilde{\pi}_{C(x)}$, and the $(x, C(x))$ -th entry of $O \text{diag}(\tilde{\pi})^{1/2}$ has value $O_{x,C(x)} \times \sqrt{\tilde{\pi}_{C(x)}}$. Thus the $(x, C(x))$ -th entry of A is

$$A_{x,C(x)} = \frac{O_{x,C(x)} \sqrt{\tilde{\pi}_{C(x)}}}{\sqrt{O_{x,C(x)} \tilde{\pi}_{C(x)}}} = \sqrt{O_{x,C(x)}}$$

The columns of A have disjoint supports since A has the same sparsity pattern as O . Furthermore, the l_2 (Euclidean) norm of any column of A is the l_1 norm of the corresponding column of O . This implies $A^\top A = I_{m \times m}$ \square

Now we give a proof of the main theorem.

Proof of Theorem 5.1. The orthogonal projection matrix onto $\text{range}(\Omega)$ is given by UU^\top and also by $\Omega(\Omega^\top \Omega)^+ \Omega^\top$. Hence from Lemma 5.1 and 5.2, we have

$$\begin{aligned} UU^\top &= \Omega(\Omega^\top \Omega)^+ \Omega^\top \\ &= (A\tilde{A}^\top)(\tilde{A}A^\top A\tilde{A}^\top)^+ (A\tilde{A}^\top)^\top \\ &= (A\tilde{A}^\top)(\tilde{A}\tilde{A}^\top)^+ (A\tilde{A}^\top)^\top = \Pi \Pi^\top \end{aligned}$$

where $\Pi = \tilde{A}(\tilde{A}^\top \tilde{A})^+ \tilde{A}^\top$ is the orthogonal projection matrix onto $\text{range}(\tilde{A})$. But since \tilde{A} has rank m , $\text{range}(\tilde{A}) = \mathbb{R}^m$ and thus $\Pi = I_{m \times m}$. Then we have $UU^\top = AA^\top$ where both U and A have orthogonal columns (Lemma 5.2). This implies that there is an orthogonal matrix $Q \in \mathbb{R}^{m \times m}$ such that $U = AQ^\top$. \square

5.3 ESTIMATION FROM SAMPLES

In Figure 4, we give an algorithm for computing an estimate of M from a sample of words $(x_1, \dots, x_N) \in [n]^N$ (where M is described in Section 5.1). The algorithm estimates unigram and bigram word probabilities u, v, B to form a plug-in estimate $\hat{\Omega}$ of Ω (defined in Eq. (1)), computes a low-rank SVD of a sparse matrix, and normalizes the rows of the resulting left singular vector matrix.

The following theorem implies the consistency of our algorithm, assuming the consistency of $\hat{\Omega}$.

Theorem 5.2. *Let $\varepsilon := \|\hat{\Omega} - \Omega\|/\sigma_m(\Omega)$, where $\sigma_m(\Omega)$ is the m -th largest singular value of Ω . If $\varepsilon \leq 0.07 \min_{x \in [n]} \{O_{x,C(x)}^{1/2}\}$, then the word embedding $f : x \mapsto \hat{M}_x$ (where \hat{M}_x is the x -th row of \hat{M}) satisfies the following property: for all $x, x', x'' \in [n]$,*

$$\begin{aligned} C(x) &= C(x') \neq C(x'') \\ \implies \|f(x) - f(x')\| &< \|f(x) - f(x'')\|; \end{aligned}$$

Input: sequence of $N \geq 2$ words $(x_1, \dots, x_N) \in [n]^N$; number of clusters m ; smoothing parameter κ .

Output: matrix $\hat{M} \in \mathbb{R}^{n \times m}$ defining $f : x \mapsto \hat{M}_x \forall x \in [n]$.

1. Compute $\hat{B} \in \mathbb{R}^{n \times n}$, $\hat{u} \in \mathbb{R}^n$, and $\hat{v} \in \mathbb{R}^n$ where

$$\hat{B}_{x,x'} := \frac{1}{N-1} \sum_{i=1}^{N-1} [[x_i = x, x_{i+1} = x']] \quad \forall x, x' \in [n]$$

$$\hat{u}_x := \frac{1}{N-1} \sum_{i=1}^{N-1} [[x_i = x]] + \frac{\kappa}{N-1} \quad \forall x \in [n]$$

$$\hat{v}_x := \frac{1}{N-1} \sum_{i=1}^{N-1} [[x_{i+1} = x]] + \frac{\kappa}{N-1} \quad \forall x \in [n]$$

2. Compute rank- m SVD of the sparse matrix

$$\hat{\Omega} := \text{diag}(\hat{u})^{-1/2} \hat{B} \text{diag}(\hat{v})^{-1/2}.$$

Let $\hat{U} \in \mathbb{R}^{n \times m}$ be a matrix of m left singular vectors of $\hat{\Omega}$ corresponding to the m largest singular values.

3. Let \hat{M} be the result of normalizing every row of \hat{U} to have length 1.

Figure 4: Estimation of M from samples.

(i.e., the embedding of any word x is closer to that of other words x' from the same cluster than it is to that of any word x'' from a different cluster).

The property established by Theorem 5.2 (proved in the appendix) allows many distance-based clustering algorithms to recover the correct clustering (e.g., single-linkage, average-linkage; see Balcan *et al.*, 2008). Moreover, it is possible to establish the finite sample complexity bounds for the estimation error of $\hat{\Omega}$ (and we do so for a simplified scenario in the (supplementary) Appendix C).

In practice, it is important to regularize the estimates \hat{u} and \hat{v} using a smoothing parameter $\kappa \geq 0$. This can be viewed as adding pseudocounts to alleviate the noise from infrequent words, and has a significant effect on the resulting representations. The practical importance of smoothing is also seen in previous methods using CCA (Cohen *et al.*, 2013; Hardoon *et al.*, 2004).

Another practical consideration is the use of richer context. So far, the context used for the token X_I is just the next token X_{I+1} ; hence, the spectral estimation is based just on unigram and bigram probabilities. However, it is straightforward to generalize the technique to use other context—details are in the appendix. For instance, if we use the previous and next tokens (X_{I-1}, X_{I+1}) as context, then we form $\hat{\Omega} \in \mathbb{R}^{n \times 2n}$ from $\hat{B} \in \mathbb{R}^{n \times 2n}$, $\hat{u} \in \mathbb{R}^n$, $\hat{v} \in \mathbb{R}^{2n}$; however, we still extract $\hat{M} \in \mathbb{R}^{n \times m}$ from $\hat{\Omega}$ in the same way to form the word embedding.

Input: vectors $\mu^{(1)}, \dots, \mu^{(n)} \in \mathbb{R}^m$ corresponding to word types $[n]$ ordered in decreasing frequency.

Output: hierarchical clustering of the input vectors.

Tightening: Given a set of clusters \mathcal{C} , the subroutine $\text{tighten}(c)$ for $c \in \mathcal{C}$ consists of the following three steps:

$$\text{nearest}(c) := \arg \min_{c' \in \mathcal{C}: c' \neq c} d(c, c')$$

$$\text{lowerbound}(c) := \min_{c' \in \mathcal{C}: c' \neq c} d(c, c')$$

$$\text{tight}(c) := \text{True}$$

Main body:

1. Initialize active clusters $\mathcal{C} = \{\{\mu^{(1)}\}, \dots, \{\mu^{(m)}\}\}$ and call $\text{tighten}(c)$ for all $c \in \mathcal{C}$.
2. For $i = m + 1$ to $n + m - 1$:
 - (a) If $i \leq n$: let $c := \{\mu^{(i)}\}$, call $\text{tighten}(c)$, and let $\mathcal{C} := \mathcal{C} \cup \{c\}$.
 - (b) Let $c^* := \arg \min_{c \in \mathcal{C}} \text{lowerbound}(c)$.
 - (c) While $\text{tight}(c^*)$ is False,
 - i. Call $\text{tighten}(c^*)$.
 - ii. Let $c^* := \arg \min_{c \in \mathcal{C}} \text{lowerbound}(c)$.
 - (d) Merge c^* and $\text{nearest}(c^*)$ in \mathcal{C} .
 - (e) For each $c \in \mathcal{C}$: if $\text{nearest}(c) \in \{c^*, \text{nearest}(c^*)\}$, set $\text{tight}(c) := \text{False}$.

Figure 5: Variant of Ward’s algorithm from Section 5.4.

5.4 AGGLOMERATIVE CLUSTERING

As established in Theorem 5.2, the word embedding obtained by mapping words to their corresponding rows of \hat{M} permits distance-based clustering algorithms to recover the correct clustering. However, with small sample sizes and model approximation errors, the property from Theorem 5.2 may not hold exactly. Therefore, we propose to compute a hierarchical clustering of the word embeddings, with the goal of finding the correct clustering (or at least a good clustering) as some pruning of the resulting tree. Simple agglomerative clustering algorithms can provably recover the correct clusters when idealized properties (such as that from Theorem 5.2) hold (Balcan *et al.*, 2008), and can also be seen to be optimizing a sensible objective regardless (Dasgupta and Long, 2005). These algorithms also yield a hierarchy of word types—just as the original Brown clustering algorithm.

We use a form of average-linkage agglomerative clustering called Ward’s algorithm (Ward Jr, 1963), which is particularly suited for hierarchical clustering in Euclidean spaces. In this algorithm, the cost of merging clusters c and c' is defined as

$$d(c, c') = \frac{|c||c'|}{|c| + |c'|} \|\mu_c - \mu_{c'}\|^2 \quad (2)$$

where $|c|$ refers to the number of elements in cluster c and $\mu_c = |c|^{-1} \sum_{u \in c} u$ is the mean of cluster c . The algorithm starts with every point (word) in its own cluster, and repeat-

edly merges the two clusters with cheapest merge cost.

Figure 5 sketches a variant of Ward’s algorithm that only considers merges among (at most) $m + 1$ clusters at a time. The initial $m + 1$ (singleton) clusters correspond to the $m + 1$ most frequent words (according to \hat{u}); after a merge, the next most frequent word (if one exists) is used to initialize a new singleton cluster. This heuristic is also adopted by the original Brown algorithm, and is known to be very effective.

Using an implementation trick from Franti *et al.* (2000), the runtime of the algorithm is $O(\gamma nm^2)$, where γ is a data-dependent constant often much smaller than m , as opposed to $O(nm^3)$ in a naive implementation in which we search for the closest pair among $O(m^2)$ pairs at every merge.

The basic idea of Franti *et al.* (2000) is the following. For each cluster, we keep an estimation of the lower bound on the distance to the nearest cluster. We also track if this lower bound is tight; in the beginning, every bound is tight. When searching for the nearest pair, we simply look for a cluster with the smallest lower bound among m clusters instead of $O(m^2)$ cluster pairs. If the cluster has a tight lower bound, we merge it with its nearest cluster. Otherwise, we tighten its bound and again look for a cluster with the smallest bound. Thus γ is the effective number of searches at each iteration. At merge, the bound of a cluster whose nearest cluster is either of the two merged clusters becomes loose. We report empirical values of γ in our experimental study (see Table 3).

6 EXPERIMENTS

To evaluate the effectiveness of our approach, we used the clusters from our algorithm as additional features in supervised models for NER. We then compared the improvement in performance and also the time required to derive the clusters against those of the Brown clustering algorithm. Additionally, we examined the mutual information (MI) of the derived clusters on the training corpus:

$$\sum_{c,c'} \frac{\text{count}(c, c')}{N} \log \frac{\text{count}(c, c')N}{\text{count}(c)\text{count}(c')} \quad (3)$$

where N is the number of tokens in the corpus, $\text{count}(c)$ is the number of times cluster c appears, and $\text{count}(c, c')$ is the number of times clusters c, c' appear consecutively. Note that this is the quantity the Brown algorithm directly maximizes (Brown *et al.*, 1992).

6.1 EXPERIMENTAL SETTINGS

For NER experiments, we used the scripts provided by Turian *et al.* (2010). We used the greedy perceptron for NER experiments (Ratinov and Roth, 2009) using the standard features as our baseline models. We used the CoNLL

Table 1: Performance gains in NER.

	vocab	context	dev	test
Baseline	—	—	90.03	84.39
Spectral	50k	LR1	92	86.72
($\kappa = 200$)	300k	LR2	92.31	87.76
Brown	50k	—	92	88.56
	300k		92.68	88.76

Table 2: Mutual information computed as in Eq. (3) on the RCV1 corpus.

	vocab size	context	MI
Spectral	50k	LR2	1.48
($\kappa = 5000$)	300k	LR2	1.54
Brown	50k	—	1.52
	300k	—	1.6

2003 dataset for NER with the standard train/dev/test split.

For the choice of unlabeled text data, we used the Reuters-RCV1 corpus which contains 205 million tokens with 1.6 million distinct word types. To keep the size of the vocabulary manageable and also to reduce noise from infrequent words, we used only a selected number of the most frequent word types and replaced all other types in the corpus with a special token. For the size of the vocabulary, we used 50,000 and 300,000.

Our algorithm can be broken down into two stages: the SVD stage (Figure 4) and the clustering stage (Figure 5). In the SVD stage, we need to choose the number of clusters m and the smoothing parameter κ . As mentioned, we can easily define Ω to incorporate information beyond one word to the right. We experimented with the following configurations for context:

1. R1 ($\Omega \in \mathbb{R}^{n \times n}$): 1 word to the right. This is the version presented in Figure 4.
2. LR1 ($\Omega \in \mathbb{R}^{n \times 2n}$): 1 word to the left/right.
3. LR2 ($\Omega \in \mathbb{R}^{n \times 4n}$): 2 words to the left/right.

6.2 COMPARISON TO THE BROWN ALGORITHM: QUALITY

There are multiple ways to evaluate the quality of clusters. We considered the improvement in the F1 score in NER from using the clusters as additional features. We also examined the MI on the training corpus. For all experiments in this section, we used 1,000 clusters for both the spectral algorithm (i.e., $m = 1000$) and the Brown algorithm.

6.2.1 NER

In NER, there is significant improvement in the F1 score from using the clusters as additional features (Table 1).

Table 3: Speed and performance comparison with the Brown algorithm for different numbers of clusters and vocabulary sizes. In all the reported runtimes, we exclude the time to read and write data. We report the F1 scores on the NER dev set; for the spectral algorithm, we report the best scores.

m	vocab	Spectral runtime				Brown runtime	Ratio (%)	Spectral F1	Brown F1
		γ	SVD	cluster	total				
200	50k	3.35	4m24s	13s	4m37s	10m37s	43.48	91.53	90.79
400		5.17	6m39s	1m8s	7m47s	37m16s	20.89	91.73	91.21
600		9.80	5m29s	3m1s	8m30s	1h33m55s	9.05	91.68	91.79
800		12.64	9m26s	6m59s	16m25s	2h20m40s	11.67	91.81	91.83
1000		12.68	11m10s	10m25s	21m35s	3h37m	9.95	92.00	92.00
1000	300k	13.77	59m38s	1h4m37s	2h4m15s	22h19m37s	9.28	92.31	92.68

The dev F1 score is improved from 90.03 to 92 with either spectral or Brown clusters using 50k vocabulary size; it is improved to 92.31 with the spectral clusters and to 92.68 with the Brown clusters using 300k vocabulary size. The spectral clusters are a little behind the Brown clusters in the test set results. However, we remark that the well-known discrepancy between the dev set and the test set in the CoNLL 2003 dataset makes a conclusive interpretation difficult. For example, Turian *et al.* (2010) report that the F1 score using the embeddings of Collobert and Weston (2008) is higher than the F1 score using the Brown clusters on the dev set (92.46 vs 92.32) but lower on the test set (87.96 vs 88.52).

6.2.2 MI

Table 2 shows the MI computed as in Eq. (3) on the RCV1 corpus. The Brown algorithm optimizes the MI directly and generally achieves higher MI scores than the spectral algorithm. However, the spectral algorithm also achieves a surprisingly respectable level of MI scores even though the MI is not its objective. That is, the Brown algorithm specifically merges clusters in order to maximize the MI score in Eq. (3). In contrast, the spectral algorithm first recovers the model parameters using SVD and perform hierarchical clustering according to the parameter estimates, without any explicit concern for the MI score.

6.3 COMPARISON TO THE BROWN ALGORITHM: SPEED

To see the runtime difference between our algorithm and the Brown algorithm, we measured how long it takes to extract clusters from the RCV1 corpus for various numbers of clusters. In all the reported runtimes, we exclude the time to read and write data. We report results with 200, 400, 600, 800, and 1,000 clusters. All timing experiments were done on a machine with dual-socket, 8-core, 2.6GHz Intel Xeon E5-2670 (Sandy Bridge). The implementations for both algorithms were written in C++. The spectral algorithm also made use of Matlab for matrix calculations such as the SVD calculation.

Table 3 shows the runtimes required to extract these clusters as well as the F1 scores on the NER dev set obtained with these clusters. The spectral algorithm is considerably faster than the Brown algorithm while providing comparable improvement in the F1 scores. The runtime difference becomes more prominent as the number of clusters increases. Moreover, the spectral algorithm scales much better with larger vocabulary size. With 1,000 clusters and 300k vocabulary size, the Brown algorithm took over 22 hours whereas the spectral algorithm took 2 hours, 4 minutes, and 15 seconds—less than 10% of the time the Brown algorithm takes.

We also note that for the Brown algorithm, the improvement varies significantly depending on how many clusters are used; it is 0.76 with 200 clusters but 1.97 with 1,000 clusters. For the spectral algorithm, this seems to be less the case; the improvement is 1.5 with 200 clusters and 1.97 with 1,000 clusters.

6.3.1 Discussion on Runtimes

The final asymptotic runtime is $O(N + \gamma nm^2)$ for the spectral algorithm and $O(N + nm^2)$ for the Brown algorithm, where N is the size of the corpus, n is the number of distinct word types, m is the number of clusters, and γ is a data-dependent constant. Thus it may be puzzling why the spectral algorithm is significantly faster in practice. We explicitly discuss the issue in this section.

The spectral algorithm proceeds in two stages. First, it constructs a scaled covariance matrix in $O(N)$ time and performs a rank- m SVD of this matrix. Table 3 shows that SVD scales well with the value of m and the size of the corpus.

Second, the algorithm performs hierarchical clustering in $O(\gamma nm^2)$ time. This stage consists of $O(\gamma nm)$ calls to an $O(m)$ time function that computes Eq. (2), that is,

$$d(c, c') = \frac{|c||c'|}{|c| + |c'|} \|\mu_c - \mu_{c'}\|^2$$

This function is quite simple: it calculates a scaled distance

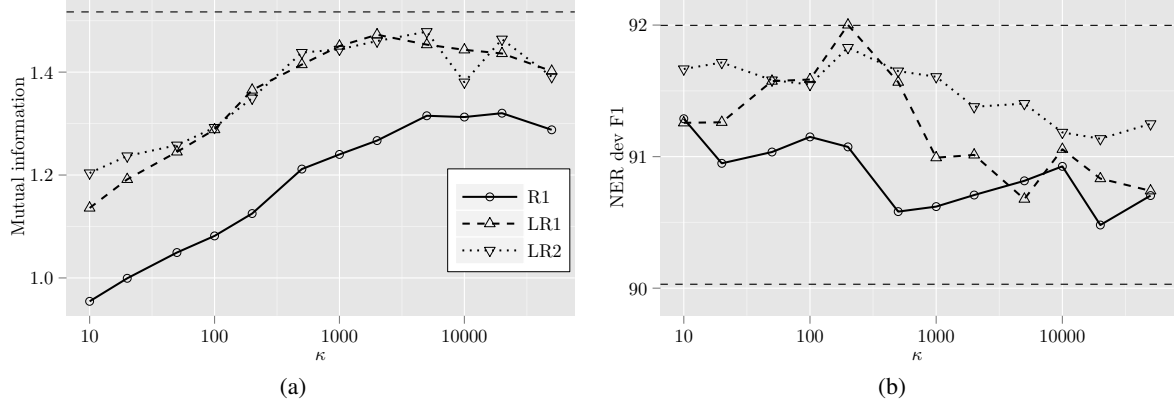


Figure 6: Effect of the choice of κ and context on (a) MI and (b) NER dev F1 score. We used 1,000 clusters on RCV1 with vocabulary size 50k. In (a), the horizontal line is the MI achieved by Brown clusters. In (b), the top horizontal line is the F1 score achieved with Brown clusters and the bottom horizontal line is the baseline F1 score achieved without using clusters.

```
computeL2usingOld(s, t, u, v, w) = L2[v][w]
- q2[v][s] - q2[s][v] - q2[w][s] - q2[s][w]
- q2[v][t] - q2[t][v] - q2[w][t] - q2[t][w]
+ (p2[v][s] + p2[w][s]) * log((p2[v][s] + p2[w][s]) / ((p1[v] + p1[w]) * p1[s]))
+ (p2[s][v] + p2[s][w]) * log((p2[s][v] + p2[s][w]) / ((p1[v] + p1[w]) * p1[s]))
+ (p2[v][t] + p2[w][t]) * log((p2[v][t] + p2[w][t]) / ((p1[v] + p1[w]) * p1[t]))
+ (p2[t][v] + p2[t][w]) * log((p2[t][v] + p2[t][w]) / ((p1[v] + p1[w]) * p1[t]))
+ q2[v][u] + q2[u][v] + q2[w][u] + q2[u][w]
- (p2[v][u] + p2[w][u]) * log((p2[v][u] + p2[w][u]) / ((p1[v] + p1[w]) * p1[u]))
- (p2[u][v] + p2[u][w]) * log((p2[u][v] + p2[u][w]) / ((p1[v] + p1[w]) * p1[u]))
```

Figure 7: A $O(1)$ function that is called $O(nm^2)$ times in Liang’s implementation of the Brown algorithm, accounting for over 40% of the runtime. Similar functions account for the vast majority of the runtime. The values in the arrays $L2, q2, p2, p1$ are precomputed. $p2[v][w] = p(v, w)$, i.e. the probability of cluster v being followed by cluster w ; $p1[v] = p(v)$ is the probability of cluster v ; $q2[v][w] = p(v, w) \log((p(v)p(w))^{-1}p(v, w))$ is the contribution of the mutual information between clusters v and w . The function recomputes $L2[v][w]$, which is the loss in log-likelihood if clusters v and w are merged. The function updates $L2$ after clusters s and t have been merged to form a new cluster u . There are many operations involved in this calculation: 6 divisions, 12 multiplications, 36 additions (26 additions and 10 subtractions), and 6 log operations.

between two vectors in \mathbb{R}^m . Moreover, it avails itself readily to existing optimization techniques such as vectorization.¹ Finally, we found that the empirical value of γ was typically small: it ranged from 3.35 to 13.77 in our experiments reported in Table 3 (higher m required higher γ).

In contrast, while the main body of the Brown algorithm requires $O(N + nm^2)$ time, the constant factors are high due to fairly complex book-keeping that is required. For example, the function in Figure 7 (obtained from Liang’s

¹Many linear algebra libraries automatically support vectorization. For instance, the Eigen library in our implementation enables vectorization by default, which gave a 2-3 time speedup in our experiments.

implementation) is invoked $O(nm^2)$ times in total: specifically, whenever two clusters s and t are merged to form a new cluster u (this happens $O(n)$ times), the function is called $O(m^2)$ times, for all pairs of clusters v, w such that v and w are not equal to s, t , or u . The function recomputes the loss in likelihood if clusters v and w are merged, after s and t are merged to form u . It requires a relatively large number of arithmetic operations, leading to high constant factors. Calls to this function alone take over 40% of the runtime for the Brown algorithm; similar functions account for the vast majority of the algorithm’s runtime. It is not clear that this overhead can be reduced.

6.4 EFFECT OF THE CHOICE OF κ AND CONTEXT

Figure 6 shows the MI and the F1 score on the NER dev set for various choices of κ and context. For NER, around 100-200 for the value of κ gives good performance. For the MI, the value of κ needs to be much larger.

LR1 and LR2 perform much better than R1 but are very similar to each other across the results, suggesting that words in the immediate vicinity are necessary and nearly sufficient for these tasks.

7 CONCLUSION

In this paper, we have presented a new and faster alternative to the Brown clustering algorithm. Our algorithm has a provable guarantee of recovering the underlying model parameters. This approach first uses SVD to consistently estimate low-dimensional representations of word types that reveal their originating clusters by exploiting the implicit disjoint-cluster assumption of the Brown model. Then agglomerative clustering is performed over these representations to build a hierarchy of word types. The resulting clusters give a competitive level of improvement in performance in NER as the clusters from the Brown algorithm,

but the spectral algorithm is significantly faster.

There are several areas for the future work. One can try to speed up the algorithm even more via a top-down rather than bottom-up approach for hierarchical clustering, for example recursively running the 2-means algorithm. Experiments with the clusters in tasks other than NER (e.g., dependency parsing), as well as larger-scale experiments, can help further verify the quality of the clusters and highlight the difference between the spectral algorithm and the Brown algorithm.

Acknowledgments

We thank Dean Foster, Matus Telgarsky, and Terry Koo for helpful discussions. This work was made possible by a research grant from Bloomberg's Knowledge Engineering team. Kim was also supported by a Samsung Scholarship.

A INCORPORATING RICHER CONTEXT

We assume a function ϕ such that for $i \in [N]$, it returns a set of positions other than i . For example, we may define $\phi(i) = \{i-1, i+1\}$ to look at one position to the left and to the right. Let $s = |\phi(i)|$ and enumerate the elements of $\phi(i)$ as j_1, \dots, j_s . Define $B^{(j)} \in \mathbb{R}^{n \times n}$, $v^{(j)} \in \mathbb{R}^n$ for all $j \in \phi(i)$ as follows:

$$\begin{aligned} B_{x,x'}^{(j)} &= P(X_i = x, X_j = x') & \forall x, x' \in [n] \\ v_x^{(j)} &= P(X_j = x) & \forall x \in [n] \end{aligned}$$

The new definitions of $B \in \mathbb{R}^{n \times ns}$, $v \in \mathbb{R}^{ns}$ are given by $B = [B^{(j_1)}, \dots, B^{(j_s)}]$ and $v = [(v^{(j_1)})^\top, \dots, (v^{(j_s)})^\top]^\top$. Letting $\Omega \in \mathbb{R}^{n \times ns}$ as in Eq. (1), it is easy to verify Theorem 5.1 using similar techniques.

B PROOF OF THEOREM 5.2

Write the rank- m SVD of Ω as $\Omega = USV^\top$, and similarly write the rank- m SVD of $\hat{\Omega}$ as $\hat{\Omega} = \hat{U}\hat{S}\hat{V}^\top$. Since Ω has rank m , it follows by Eckart-Young that

$$\|\hat{U}\hat{S}\hat{V}^\top - \hat{\Omega}\| \leq \|\Omega - \hat{\Omega}\|.$$

Therefore, by the triangle inequality,

$$\|\hat{U}\hat{S}\hat{V}^\top - USV^\top\| \leq 2\|\Omega - \hat{\Omega}\| = 2\varepsilon\sigma_m(\Omega).$$

This implies, via applications of Wedin's theorem and Weyl's inequality,

$$\|U_\perp^\top \hat{U}\| \leq 2\varepsilon \quad \text{and} \quad \|\hat{U}_\perp^\top U\| \leq \frac{2\varepsilon}{1-2\varepsilon} \quad (4)$$

where $U_\perp \in \mathbb{R}^{n \times (n-m)}$ is a matrix whose columns form an orthonormal basis for the orthogonal complement of the

range of U , and $\hat{U}_\perp \in \mathbb{R}^{n \times (n-m)}$ is similarly defined (and note that $\varepsilon < 1/2$ by assumption).

Recall that by Theorem 5.1, there exists an orthogonal matrix $Q \in \mathbb{R}^{m \times m}$ such that $U = \sqrt{O}Q^\top$. Define $\hat{Q} := \hat{U}^\top \sqrt{O} = \hat{U}^\top UQ$, and, for all $c \in [m]$, $\hat{q}_c := \hat{Q}e_c$. The fact that $\|UQe_c\| = 1$ implies

$$\|\hat{q}_c\| = \sqrt{1 - \|\hat{U}_\perp \hat{U}_\perp^\top UQe_c\|^2} \leq 1.$$

Therefore, by Eq. (4),

$$1 \geq \|\hat{q}_c\| \geq \|\hat{q}_c\|^2 \geq 1 - \left(\frac{2\varepsilon}{1-2\varepsilon}\right)^2. \quad (5)$$

We also have, for $c \neq c'$,

$$\hat{q}_c^\top \hat{q}_{c'} \leq \|\hat{U}_\perp^\top UQe_c\| \|\hat{U}_\perp^\top UQe_{c'}\| \leq \left(\frac{2\varepsilon}{1-2\varepsilon}\right)^2, \quad (6)$$

where the first inequality follows by Cauchy-Schwarz, and the second inequality follows from (4). Therefore, by Eq. (5) and Eq. (6), we have for $c \neq c'$,

$$\|\hat{q}_c - \hat{q}_{c'}\|^2 \geq 2 \left(1 - 2 \left(\frac{2\varepsilon}{1-2\varepsilon}\right)^2\right). \quad (7)$$

Let $\bar{o}_x := O_{x,C(x)}^{1/2}$. Recall that $\sqrt{O}^\top e_x = \bar{o}_x e_{C(x)} \in \mathbb{R}^m$, so $\hat{Q}\sqrt{O}^\top e_x = \bar{o}_x \hat{q}_{C(x)}$ and $\|\hat{Q}\sqrt{O}^\top e_x\| = \bar{o}_x \|q_{C(x)}\|$. By the definition of \hat{Q} , we have

$$\hat{U} - \sqrt{O}\hat{Q}^\top = \hat{U} - UU^\top \hat{U} = U_\perp U_\perp^\top \hat{U}$$

This implies, for any $x \in [n]$,

$$\begin{aligned} \|\hat{U}^\top e_x - \bar{o}_x \hat{q}_{C(x)}\| &= \|(\hat{U} - \sqrt{O}\hat{Q}^\top)^\top e_x\| \\ &= \|\hat{U}^\top U_\perp U_\perp^\top e_x\| \leq 2\varepsilon \end{aligned} \quad (8)$$

by Eq. (4). Moreover, by the triangle inequality,

$$\|\|\hat{U}^\top e_x\| - \bar{o}_x \|q_{C(x)}\|\| \leq 2\varepsilon. \quad (9)$$

Since $\hat{M}^\top e_x = \|\hat{U}^\top e_x\|^{-1} \hat{U}^\top e_x$, we have

$$\begin{aligned} \|\hat{M}^\top e_x - \hat{q}_{C(x)}\| &= \left\| \frac{1}{\|\hat{U}^\top e_x\|} \hat{U}^\top e_x - \hat{q}_{C(x)} \right\| \\ &\leq \frac{1}{\bar{o}_x} \|\hat{U}^\top e_x - \bar{o}_x \hat{q}_{C(x)}\| + |1 - \|\hat{q}_{C(x)}\|| \\ &\quad + \frac{|\bar{o}_x \|\hat{q}_{C(x)}\| - \|\hat{U}^\top e_x\||}{\bar{o}_x} \\ &\leq \frac{4\varepsilon}{\bar{o}_x} + \left(\frac{2\varepsilon}{1-2\varepsilon}\right)^2, \end{aligned} \quad (10)$$

where the first inequality follow by the triangle inequality and norm homogeneity, and the second inequality uses Eq. (8), Eq. (9), and Eq. (5). Using Eq. (10), we may upper bound the distance $\|\hat{M}^\top e_x - \hat{M}^\top e_{x'}\|$ when $C(x) = C(x')$; using Eq. (7) and Eq. (10), we may lower bound the distance $\|\hat{M}^\top e_x - \hat{M}^\top e_{x''}\|$ when $C(x) \neq C(x'')$. The theorem then follows by invoking the assumption on ε .

References

- Ando, R. K. and Zhang, T. (2005). A framework for learning predictive structures from multiple tasks and unlabeled data. *The Journal of Machine Learning Research*, **6**, 1817–1853.
- Balcan, M.-F., Blum, A., and Vempala, S. (2008). A discriminative framework for clustering via similarity functions. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, pages 671–680.
- Brown, P. F., Desouza, P. V., Mercer, R. L., Pietra, V. J. D., and Lai, J. C. (1992). Class-based n-gram models of natural language. *Computational linguistics*, **18**(4), 467–479.
- Cohen, S. B., Stratos, K., Collins, M., Foster, D. P., and Ungar, L. (2013). Experiments with spectral learning of latent-variable pcfgs. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.
- Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.
- Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, **12**, 2493–2537.
- Dasgupta, S. and Long, P. M. (2005). Performance guarantees for hierarchical clustering. *J. Comput. Syst. Sci.*, **70**(4), 555–569.
- Dhillon, P. S., Foster, D., and Ungar, L. (2011). Multi-view learning of word embeddings via CCA. In *Advances in Neural Information Processing Systems (NIPS)*, volume 24.
- Franti, P., Kaukoranta, T., Shen, D.-F., and Chang, K.-S. (2000). Fast and memory efficient implementation of the exact pnn. *Image Processing, IEEE Transactions on*, **9**(5), 773–777.
- Gao, J., Goodman, J., Miao, J., *et al.* (2001). The use of clustering techniques for language modeling—application to asian languages. *Computational Linguistics and Chinese Language Processing*, **6**(1), 27–60.
- Hardoon, D. R., Szedmak, S., and Shawe-Taylor, J. (2004). Canonical correlation analysis: An overview with application to learning methods. *Neural Computation*, **16**(12), 2639–2664.
- Hotelling, H. (1936). Relations between two sets of variates. *Biometrika*, **28**(3/4), 321–377.
- Hsu, D., Kakade, S. M., and Zhang, T. (2012). A spectral algorithm for learning hidden markov models. *Journal of Computer and System Sciences*, **78**(5), 1460–1480.
- Kneser, R. and Ney, H. (1993). Improved clustering techniques for class-based statistical language modelling. In *Third European Conference on Speech Communication and Technology*.
- Koo, T., Carreras, X., and Collins, M. (2008). Simple semi-supervised dependency parsing. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.
- Liang, P. (2005). *Semi-Supervised Learning for Natural Language*. Master’s thesis, Massachusetts Institute of Technology.
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013a). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.
- Mikolov, T., Yih, W.-t., and Zweig, G. (2013b). Linguistic regularities in continuous space word representations. In *Proceedings of NAACL-HLT*, pages 746–751.
- Miller, S., Guinness, J., and Zamanian, A. (2004). Name tagging with word clusters and discriminative training. In *HLT-NAACL*, volume 4, pages 337–342. Citeseer.
- Ng, A. Y., Jordan, M. I., Weiss, Y., *et al.* (2002). On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, **2**, 849–856.
- Pereira, F., Tishby, N., and Lee, L. (1993). Distributional clustering of english words. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, pages 183–190. Association for Computational Linguistics.
- Ratinov, L. and Roth, D. (2009). Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155. Association for Computational Linguistics.
- Shanbehzadeh, J. and Ogunbona, P. (1997). On the computational complexity of the lbg and pnn algorithms. *Image Processing, IEEE Transactions on*, **6**(4), 614–616.
- Tropp, J. A. (2011). User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, pages 1–46.
- Turian, J., Ratinov, L., and Bengio, Y. (2010). Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics.
- Ward Jr, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, **58**(301), 236–244.

Inference Complexity in Continuous Time Bayesian Networks

Liessman Sturlaugson and John W. Sheppard

Department of Computer Science

Montana State University

Bozeman, MT 59717

{liessman.sturlaugson, john.sheppard}@cs.montana.edu

Abstract

The continuous time Bayesian network (CTBN) enables temporal reasoning by representing a system as a factored, finite-state Markov process. The CTBN uses a traditional Bayesian network (BN) to specify the initial distribution. Thus, the complexity results of Bayesian networks also apply to CTBNs through this initial distribution. However, the question remains whether propagating the probabilities through time is, by itself, also a hard problem. We show that exact and approximate inference in continuous time Bayesian networks is NP-hard even when the initial states are given.

1 INTRODUCTION

Currently, little theoretical work has been done on the complexity of inference in CTBNs. Most of what is known about the complexity of CTBNs is derived from the fact that a Bayesian network is used to specify the initial distribution. However, despite the similarity in the names, inference in CTBNs is different than inference in Bayesian networks because the CTBN must reason over continuous time. The question becomes whether this problem of calculating evolving probabilities through time is also NP-hard, independent of the initial distribution.

The only known exact inference procedure for CTBNs is exponential in the number of nodes, but this does not necessarily imply that there does not exist an alternative approach for exact inference that performs in polynomial time even for the worst case. Furthermore, what expectations might we have about the various approximate inference algorithms that have been developed for CTBNs? Does the accuracy vs. complexity trade-off of these approximation algorithms avoid the NP-hardness of their discrete-time counterparts? In

this work, we prove that exact and approximate inference in CTBNs are both NP-hard, even when the initial states are given. Thus, the complexity is also in reasoning over the factored Markov process, not just in reasoning over the Bayesian network for determining the probabilities for the initial states.

2 BACKGROUND

To place our work in context, we begin by presenting background information on Bayesian networks, dynamic Bayesian networks, and then CTBNs.

2.1 BAYESIAN NETWORKS

Bayesian networks are probabilistic graphical models that use nodes and arcs in a directed acyclic graph to represent a joint probability distribution over a set of variables (Koller & Friedman, 2009). Let $P(\mathbf{X})$ be a joint probability distribution over n variables $X_1, \dots, X_n \in \mathbf{X}$. A Bayesian network \mathcal{B} is a directed, acyclic graph in which each variable X_i is represented by a node in the graph. Let $\mathbf{Pa}(X_i)$ denote the parents of node X_i in the graph. The graph representation of \mathcal{B} factors the joint probability distribution as:

$$P(\mathbf{X}) = \prod_{i=1}^n P(X_i | \mathbf{Pa}(X_i)).$$

2.2 DYNAMIC BAYESIAN NETWORKS

The traditional Bayesian network is a static model. However, we can introduce the concept of time (or at least sequence) into the network by assigning discrete timesteps to the nodes to create a dynamic Bayesian network.

A dynamic Bayesian network (DBN) is a type of Bayesian network that uses a series of connected timesteps, each of which contains a copy of a Bayesian network \mathbf{X}_t indexed by timestep t . The probability

distribution of a variable at a given timestep can be conditionally dependent on states of that variable or other variables throughout previous timesteps. For first-order DBNs, dependence does not go further than the immediately previous timestep. Therefore, the joint probability distribution for a first-order DBN of T timesteps factors as:

$$P(\mathbf{X}_0, \dots, \mathbf{X}_{T-1}) = P(\mathbf{X}_0) \prod_{t=0}^{T-1} P(\mathbf{X}_{t+1} | \mathbf{X}_t).$$

2.3 CONTINUOUS TIME BAYESIAN NETWORKS

As can be seen from the preceding section, the DBN is restricted to discrete timesteps. The CTBN avoids discretizing time by using conditional Markov processes instead of conditional probability tables. We now formally define the CTBN and then survey its inference algorithms and applications.

2.3.1 CTBN Definition

Let \mathbf{X} be a set of Markov processes $\{X_1, X_2, \dots, X_n\}$, where each process X_i has a finite number of discrete states. A continuous time Bayesian network is a tuple $\mathcal{N} = \langle \mathcal{B}, \mathcal{C} \rangle$. The Bayesian network \mathcal{B} has nodes corresponding to \mathbf{X} and is used only for determining $P(\mathbf{X}_0)$, the initial distribution of the process. Evidence at the initial time ($t = 0$) is incorporated by setting evidence in \mathcal{B} and performing Bayesian network inference. The continuous-time transition model \mathcal{C} describes the evolution of the process from this initial distribution and is specified as:

- A directed (possibly cyclic) graph \mathcal{G} with nodes X_1, X_2, \dots, X_n , where $\mathbf{Pa}(X_i)$ denotes the parents of X_i in \mathcal{G} ,
- A set of conditional intensity matrices (CIMs) $\mathbf{A}_{X|\mathbf{Pa}(X)}$ associated with X for each possible combination of state instantiations of $\mathbf{Pa}(X)$.

Each conditional intensity matrix $\mathbf{A}_{X|\mathbf{Pa}(X)}$ gives the dynamics of node X when the states of $\mathbf{Pa}(X)$ are fixed. Each entry $a_{i,j} \geq 0$, $i \neq j$ gives the transition intensity of the node moving from state i to state j , and each entry $a_{i,i} < 0$ controls the amount of time the node remains in state i . With the diagonal entries constrained to be non-positive, the probability density function for the node remaining in state i is given by $|a_{i,i}| \exp(a_{i,i}t)$, with t being the amount of time spent in state i , making the probability of remaining in a state decrease exponentially with respect to time. The expected sojourn time for state i is $1/|a_{i,i}|$. Each row is constrained to sum to zero, $\sum_j a_{i,j} = 0 \forall i$, meaning

that the transition probabilities from state i can be calculated as $a_{i,j}/|a_{i,i}| \forall j, i \neq j$.

2.3.2 CTBN Inference Algorithms

The only exact inference algorithm that exists so far for CTBNs combines all of the conditional intensity matrices into the single full joint intensity matrix, with states as the Cartesian product of all of the node's states (Fan, Xu, & Shelton, 2010). Thus, the size of this matrix is exponential in the number of nodes and the number of states. Because this method ignores the factored nature of the network, research on CTBN inference has focused exclusively on approximation algorithms.

Expectation propagation (Nodelman, Koller, & Shelton, 2005; Saria, Nodelman, & Koller, 2007) has been developed for CTBNs, in which neighboring nodes employ a message-passing scheme for each interval of evidence. The messages are approximate "marginals," a projection of a node's conditional intensity matrix onto a single, approximating unconditional intensity matrix. Messages are continually passed until all of the nodes have a consistent distribution over the interval of evidence.

There have been a number of sample-based inference algorithms developed for CTBNs, including importance sampling (Fan et al., 2010; Fan & Shelton, 2008; Weiss, Natarajan, & Page, 2013) and Gibbs sampling (El-Hay, Friedman, & Kupferman, 2008; Rao & Teh, 2013). Importance sampling answers queries from a set of weighted samples that are generated in conformance to the evidence. The weight of each sample is the likelihood of the sample given the evidence. Gibbs sampling, by contrast, is a sampling procedure that takes a Markov Chain Monte Carlo (MCMC) approach. For each variable over each interval of evidence, the states in the Markov blanket (that is, the node's parents, children, and children's parents) are held constant and a random walk is performed on the state of the node. After sufficient sampling, the distribution of the random walk will converge to the true distribution for that interval of evidence.

Methods using variational techniques, such as mean-field approximation (Cohn, 2009; Cohn, El-Hay, Friedman, & Kupferman, 2009) and belief propagation (El-Hay, Cohn, Friedman, & Kupferman, 2010) have also been developed. These methods propagate the products of inhomogeneous Markov processes to approximate the distribution using systems of ordinary differential equations.

2.3.3 CTBN Applications

CTBNs have found use in a wide variety of temporal applications. For example, CTBNs have been used for inferring users' presence, activity, and availability over time (Nodelman & Horvitz, 2003); robot monitoring (Ng, Pfeffer, & Dearden, 2005); modeling server farm failures (Herbrich, Graepel, & Murphy, 2007); modeling social network dynamics (Fan & Shelton, 2009; Fan, 2009); modeling sensor networks (Shi, Tang, & You, 2010); building intrusion detection systems (Xu & Shelton, 2010; Xu, 2010; Xu & Shelton, 2008); predicting the trajectory of moving objects (Qiao et al., 2010); and diagnosing cardiogenic heart failure and anticipating its likely evolution (Gatti, Luciani, & Stella, 2011; Gatti, 2011).

2.4 PREVIOUS COMPLEXITY RESULTS

As mentioned, most of the complexity theory surrounding CTBNs is derived from the Bayesian network for the initial distribution. However, one complexity result specific to CTBNs arises from the difference between BN and CTBN structure learning. In structure learning, it is common to assign a scoring function to arcs in the network that quantifies how well the network topology matches the training data. Nodelman (2007) gives a polynomial-time algorithm for finding the highest-scoring set of k parents for a CTBN node. The corresponding problem in a Bayesian network has been shown to be NP-hard, even for $k = 2$, due to the acyclic constraint of Bayesian networks (Chickering, 1996). Essentially, because cycles are allowed in CTBNs, each node can maximize its score independently.

Because the CTBN is relatively new, much of the complexity theory surrounding CTBNs has yet to be fully explored. This work intends to expand the complexity theory of CTBN inference. The work builds on the complexity results of BNs, which we now review.

Theorem 2.1. (Cooper, 1990) *Exact inference in Bayesian networks is NP-hard.*

Proof. Cooper proved the NP-hardness of Bayesian network inference via a reduction from 3SAT. The 3SAT problem consists of a set of m clauses $C = \{c_1, c_2, \dots, c_m\}$ made up of a finite set V of n Boolean variables. Each clause contains a disjunction of three literals over V , for example, $c_3 = (v_2 \wedge \neg v_3 \wedge v_4)$. The 3SAT problem is determining whether there exists a truth assignment for V such that all the clauses in C are satisfied.

The 3SAT problem can be reduced to a Bayesian network decision problem of whether, for a $True(T)/False(F)$ node X in the network,

$P(X = T) > 0$ or $P(X = T) = 0$. We can represent any 3SAT instance by a Bayesian network as follows. For each Boolean variable v_i in V , we add a corresponding $True(T)/False(F)$ node V_i to the network such that $P(V_i = T) = \frac{1}{2}$ and $P(V_i = F) = \frac{1}{2}$. For each clause C_j , we add a corresponding $True(T)/False(F)$ node C_j to the network as a child of the three nodes corresponding to its three Boolean variables. Let w_j be the clause corresponding to the state of the three parents of C_j , and let $eval(w_j)$ be the truth function for this clause. The conditional probabilities of the node are

$$P(C_j = T|w_j) = \begin{cases} 1 & \text{if } eval(w_j) = T \\ 0 & \text{if } eval(w_j) = F \end{cases}$$

Finally, for each clause C_k , we add a $True(T)/False(F)$ node D_k . Each D_k is conditionally dependent on C_k and on D_{k-1} (except for D_1). The conditional probabilities for D_1 are

$$P(D_1 = T|C_1) = \begin{cases} 1 & \text{if } C_1 = T \\ 0 & \text{otherwise} \end{cases}$$

Similarly, the conditional probabilities for D_k ($k > 1$) are

$$P(D_k = T|C_k, D_{k-1}) = \begin{cases} 1 & \text{if } C_k = T \wedge D_{k-1} = T \\ 0 & \text{otherwise} \end{cases}$$

Figure 1 shows the BN topology and conditional probability tables for determining the satisfiability of the clause $(v_1 \vee v_2 \vee v_3) \wedge (\neg v_1 \vee \neg v_2 \vee v_3) \wedge (v_2 \vee \neg v_3 \vee v_4)$.

Importantly, the construction of this Bayesian network is polynomial in the length of the Boolean expression. For a 3SAT instance of $|V|$ variables and $|C|$ clauses, the corresponding Bayesian network has $|V| + 2|C|$ nodes. Furthermore, each node of the Bayesian network has no more than three parents, constraining the largest conditional probability table to have no more than 16 entries, for a maximum of $2|V| + 16|C| + 8(|C| - 1) + 4$ entries for the entire network.

The probabilities of the V nodes allow for every combination of truth assignments to the Boolean variables. From there, the C and D nodes enforce the logical relations of the clauses using the Bayesian network's conditional probability tables. As such, the 3SAT instance is satisfiable if and only if $P(D_m = T) > 0$, that is, if and only if there is a non-zero probability that some instantiation of the V nodes to T and F will cause all of the clauses to be satisfied. Thus, if an algorithm exists that is able to efficiently compute the exact probabilities in arbitrary Bayesian networks, the algorithm

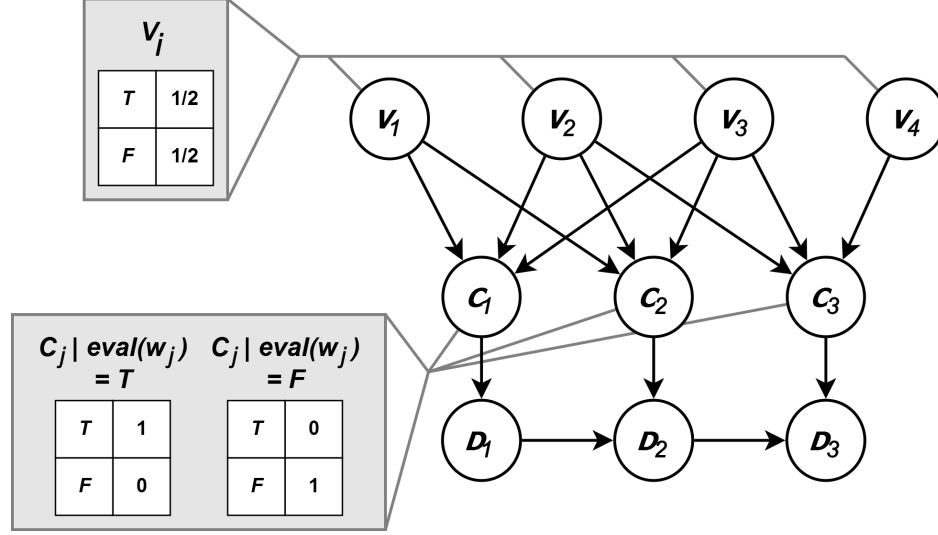


Figure 1: Network with conditional probability tables for example reduction from 3SAT to BN inference.

can efficiently decide whether $P(D_m = T) > 0$ for the specially constructed networks that can represent arbitrary instances of 3SAT. \square

Furthermore, it is known that even absolute and relative approximations in BNs is NP-hard (Dagum & Luby, 1993). These approximations are defined formally as follows. Suppose we have a real value ϵ between 0 and 1, a BN with binary-valued nodes V , and two nodes X and Y in V instantiated to x and y , respectively.

Definition 2.1. A relative approximation is an estimate $0 \leq Z \leq 1$ such that

$$\frac{P(X = x|Y = y)}{(1 + \epsilon)} \leq Z \leq P(X = x|Y = y)(1 + \epsilon).$$

Definition 2.2. An absolute approximation is an estimate $0 \leq Z \leq 1$ such that

$$P(X = x|Y = y) - \epsilon \leq Z \leq P(X = x|Y = y) + \epsilon.$$

The proof of NP-hardness for relative approximation follows directly from the proof for exact inference. Satisfiability of the clause is determined whether $Z = 0$ or $Z > 0$, which is not influenced by the choice of ϵ .

Theorem 2.2. (Dagum & Luby, 1993) Absolute approximate inference in Bayesian networks is NP-hard.

The proof of NP-hardness for absolute approximation starts with the reduction for exact inference as above, representing the variables and clauses with the same network and parameters. This time, one by one a truth assignment is set for each Boolean variable v_i , and the corresponding node V_i is removed from the

network. The truth assignment for v_i is determined by the higher probability of $P(V_i = T|D_m = T)$ and $P(V_i = F|D_m = T)$. However, if there exists an efficient approximate BN inference algorithm that can guarantee to be within $\epsilon = \frac{1}{2}$ of the exact probability on arbitrary Bayesian networks, this algorithm can be used to efficiently determine satisfying truth assignments to all Boolean variables of an arbitrary instance of 3SAT. Furthermore, any approximation with $\epsilon \geq \frac{1}{2}$ for a two-state node (the simplest case) is no better than random guessing.

These proofs are for Bayesian networks, which apply to the initial distribution of a CTBN. While the CTBN and DBN are formulated differently, Cohn, El-Hay, Friedman, and Kupferman (2010) prove that a DBN becomes asymptotically equivalent to a CTBN as the interval of time between timesteps approaches zero. One might be tempted to argue that the Bayesian network complexity proofs therefore apply to the CTBN. However, it is not always clear that moving from a discrete space to a continuous space preserves the complexity results. For example, take the difference between linear programming and integer linear programming, the former being solvable in polynomial time with the latter being NP-hard. Thus, we prove the complexity results for CTBNs explicitly.

3 EXACT INFERENCE IN CTBNS

We show that exact inference in CTBNs is NP-hard, even when given the exact initial states, following a similar reduction as the proof for BNs but using the conditional intensity matrices of the CTBN instead of the conditional probability tables. Figure 2 shows the

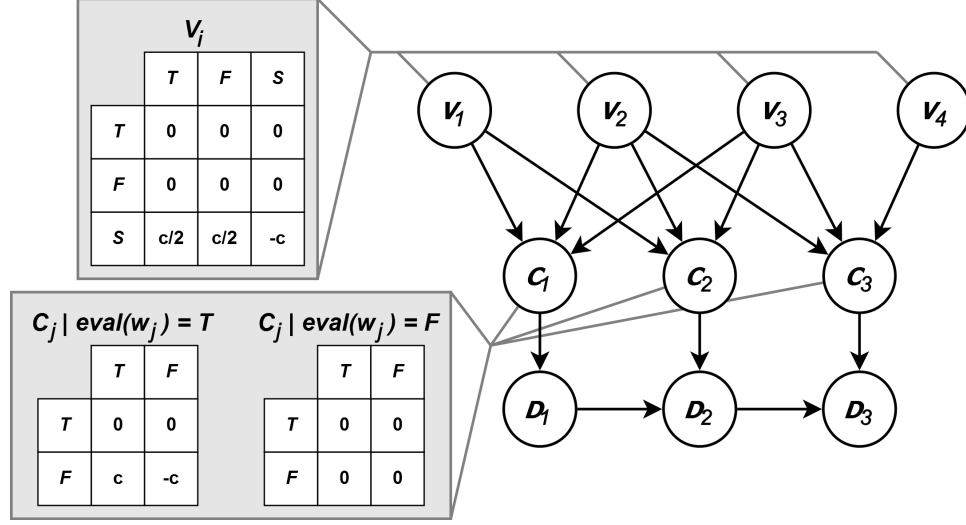


Figure 2: Network with conditional intensity matrices for example reduction from 3SAT to CTBN inference.

CTBN topology and conditional intensity matrices for determining the satisfiability of the clause $(v_1 \vee v_2 \vee v_3) \wedge (\neg v_1 \vee \neg v_2 \vee v_3) \wedge (v_2 \vee \neg v_3 \vee v_4)$.

Theorem 3.1. *Exact inference in Continuous Time Bayesian Networks is NP-hard even when given the initial states.*

Proof. The CTBN topology matches that of the BN for representing variables and clauses, but the nodes are specified differently. For each Boolean variable v_i in V , we add a corresponding three-state node V_i to the network. The three states in order are *True*(T), *False*(F), and *Start*(S), which is the initial state for node V_i . We set the unconditional intensity matrix of V_i to be

$$\mathbf{A}_{V_i} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ c/2 & c/2 & -c \end{pmatrix}$$

for some constant $c > 0$.

For each clause C_j , we add a corresponding *True*(T)/*False*(F) node C_j to the network as a child of the three nodes corresponding to its three Boolean variables. As before, let w_j be the clause corresponding to the state of the three parents of C_j , and let $eval(w_j)$ be the truth function for this clause. The function $eval$ is extended to return *False* whenever the clause w_j contains a node in state S . The conditional intensity matrices of C_j are

$$\mathbf{A}_{C_j | eval(w_j)=T} = \begin{pmatrix} 0 & 0 \\ c & -c \end{pmatrix}$$

and

$$\mathbf{A}_{C_j | eval(w_j)=F} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

We set the initial state of each C_j to be the *F* state (the second row of the matrices).

Finally, for each clause C_k , we add a *True*(T)/*False*(F) node D_k . Each D_k is conditionally dependent on C_k and on D_{k-1} (except for D_1). The conditional intensity matrices for D_k are

$$\mathbf{A}_{D_k | eval(C_k \wedge D_{k-1})=T} = \begin{pmatrix} 0 & 0 \\ c & -c \end{pmatrix}$$

and

$$\mathbf{A}_{D_k | eval(C_k \wedge D_{k-1})=F} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

As with the C_j nodes, we set the initial state of each D_k to be the *F* state.

The conditional intensity matrices of the CTBN enforce the logical constraints of the Boolean expression, replacing the conditional probability tables of the Bayesian network. As before, a 3SAT instance of $|V|$ variables and $|C|$ clauses generates $|V| + 2|C|$ nodes in the corresponding CTBN. Each node still has no more than three parents but now each intensity matrix has 9 or 4 entries, meaning that there is a maximum of $9|V| + 108|C| + 16(|C| - 1) + 8$ conditional intensity matrix entries for the entire network.

Let $D_m(t)$ be the state of D_m at time t . The 3SAT instance is satisfiable if and only if $P(D_m(t) = T) > 0$ for any time $t > 0$. Assume that the Boolean expression is satisfiable by some combination of T/F state assignments to the variables in V . The V_i nodes start in the S state at time $t = 0$. The time that each variable remains in S is exponentially distributed, after which the variables transition to either T or F with equal probability and remain in that state. Therefore, there is a non-zero probability for each combi-

nation of T/F states in the V_i 's at $t > 0$. Whenever three of these states satisfy a clause C_j , there is a non-zero probability for C_j to transition from F to T when $t > 0$. Likewise, once the parents of D_k are in T there is a non-zero probability for D_k to transition from F to T when $t > 0$. Thus, if the Boolean expression is satisfiable, there is a non-zero probability that each clause is satisfied at $t > 0$, and therefore $P(D_m(t) = T) > 0$. On the other hand, assume that the Boolean expression is not satisfiable. Then there exists some clause C_j that remains in F for all $t > 0$. Therefore, D_k will remain in F for all $k \geq j$, which means that $P(D_m(t) = T) = 0$ for all $t \geq 0$. \square

4 APPROXIMATE INFERENCE IN CTBNS

We prove similar results for approximate inference with CTBNs as well.

Theorem 4.1. *Relative approximate inference in Continuous Time Bayesian Networks is NP-hard even when given the initial states.*

Proof. Because the determination is whether $P(D_m(t) = T) = 0$ or $P(D_m(t) = T) > 0$, a relative approximation for $P(D_m(t) = T)$ with any error bound also gives a solution to the satisfiability of the Boolean expression. \square

We now turn to the absolute approximation. Because even approximate inference in BNs is NP-hard, it seems reasonable to suspect that a similar conclusion also holds for approximate inference in CTBNs. We now show how an absolute error approximation algorithm for CTBNs can be used to find a satisfying assignment to the Boolean expression or to determine that it is not satisfiable.

Theorem 4.2. *Absolute approximate inference in Continuous Time Bayesian Networks is NP-hard even when given the initial states.*

Proof. We start by assuming that the expression has at least one satisfying assignment. A satisfying truth assignment can be found one variable at a time by choosing $t > 0$ conditioning on $D_m(t) = T$. Let $t' \geq t$.

By construction, $P(V_i(t') = S | D_m(t) = T) = 0$. This is important, because it ensures that $P(V_i(t') = T | D_m(t) = T) + P(V_i(t') = F | D_m(t) = T) = 1$. Let $a \in \{T, F\}$, and let \hat{P}_a^i denote the absolute error approximation with ϵ for the probability $P(V_i(t') = a | D_m(t) = T)$. Without loss of generality, assume that V_i can be satisfied only when $a = T$. Then $P(V_i(t') = T | D_m(t) = T) = 1$ and $P(V_i(t') = F | D_m(t) = T) = 0$.

Therefore, it must be that $\hat{P}_T^i > \hat{P}_F^i$ whenever $\epsilon < \frac{1}{2}$. We compute both \hat{P}_T^i and \hat{P}_F^i and change the initial state of V_i to T if $\hat{P}_T^i > \hat{P}_F^i$ and to F otherwise. This process continues for $i = 1, \dots, |V|$ to determine truth assignments for all variables in the Boolean expression. Therefore, if there exists a polynomial-time approximation algorithm for CTBN inference with $\epsilon < \frac{1}{2}$ that can condition on evidence, it can be used to solve arbitrary instances of 3SAT in polynomial time as well. \square

5 EMPIRICAL VALIDATION

We can empirically validate these theoretical results by taking Boolean expressions and performing inference in the corresponding CTBN. Specifically, we demonstrate three Boolean expressions, listed as follows.

$$BE1 = (v_1 \vee v_2 \vee v_3) \wedge (\neg v_1 \vee \neg v_2 \vee v_3) \\ \wedge (v_2 \vee \neg v_3 \vee v_4)$$

$$BE2 = (v_1 \vee v_1 \vee v_1) \wedge (\neg v_2 \vee \neg v_2 \vee \neg v_2) \\ \wedge (v_3 \vee v_3 \vee v_3)$$

$$BE3 = (v_1 \vee v_1 \vee v_2) \wedge (v_1 \vee v_1 \vee \neg v_2) \\ \wedge (\neg v_1 \vee \neg v_1 \vee v_2) \wedge (\neg v_1 \vee \neg v_1 \vee \neg v_2)$$

Note that $BE1$ is the Boolean expression given as an example earlier and with the CTBN shown in Figure 2. A total of 10 out of its 16 possible truth assignments satisfy the expression. Note that $BE2$ has a single satisfying assignment and that $BE3$ is unsatisfiable.

To determine the satisfiability of each of these expressions using the corresponding CTBN, we performed forward sampling with 100,000 samples and $c = 100$ over the interval time $[0, 0.2)$. We queried the proportion of samples with which $D_m(t) = T$ for $t = 0$ to $t = 0.2$ in increments of 0.01. The results are shown in Figure 3. For the two satisfiable expressions, $BE1$ and $BE2$, $P(D_m(t) = T) > 0$ for $t \geq 0.01$, while for the unsatisfiable query $BE3$, $P(D_m(t) = T) = 0$ for all $t \in [0, 0.2)$.

Also note the values to which the probabilities are converging. For $BE1$, the probability ended at an estimated 0.622, whereas the proportion of satisfying assignments is $10/16 = 0.625$. For $BE2$, the probability ended at an estimated 0.127, whereas the proportion of satisfying assignments is $1/8 = 0.125$. As the number of samples increases, the probabilities converge to the proportion of satisfying assignments.

Next, we validate the approach through which an approximation of $P(V_i(t) = T | D_m(t) = T)$ is able to determine a satisfying assignment to each V_i . Because we are conditioning on evidence, we use importance sampling (Fan et al., 2010) and smooth zero entries in the unconditional intensity matrices with $\pm 10^{-6}$. We

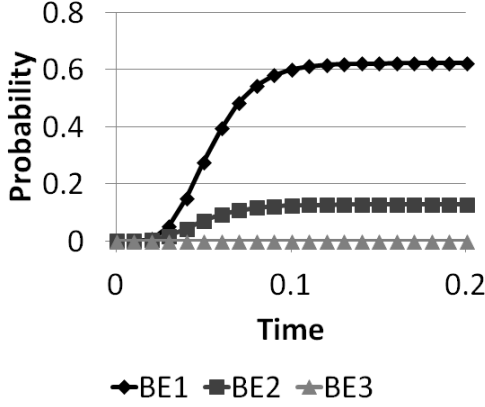


Figure 3: Empirical satisfiability results for the three Boolean expressions.

Table 1: Empirical results for approximating $P(V_i(0.2) = T | D_m(0.2) = T)$

	V_1	V_2	V_3	V_4
<i>BE1</i>	≈ 0.50	≈ 0.60	≈ 0.60	≈ 0.60
<i>BE2</i>	1	0	1	-
<i>BE3</i>	NaN	NaN	-	-

ignore samples with infinitesimal weights, as an infinitesimal weight implies that the corresponding sample contains a transition that violates the Boolean expression. The results with 100,000 samples are shown in Table 1.

The table shows that the importance sampling algorithm was correctly able to determine a satisfying truth assignments to each variable or determine that no truth assignments was possible. For *BE1*, by setting v_2 , v_3 , and v_4 to T , the Boolean expression is satisfied regardless of the value of v_1 , which is why the estimate was approximately 0.5, that is, either T or F is equally probably for satisfying the expression. For *BE2*, the importance sampling algorithm determined the single satisfying truth assignment. For *BE3*, no feasible samples could be generated because it is conditioned on an impossible event $D_m(0.2) = T$, indicating that the expression is unsatisfiable.

While we showed that we are able to solve these instances of 3SAT by CTBN sampling methods, the complexity is still exponential in the length of the Boolean expression. To demonstrate this, we show the average sample count necessary to determine the satisfiability of the Boolean expression

$$\bigwedge_{i=1,\dots,n} (v_i \vee v_i \vee v_i)$$

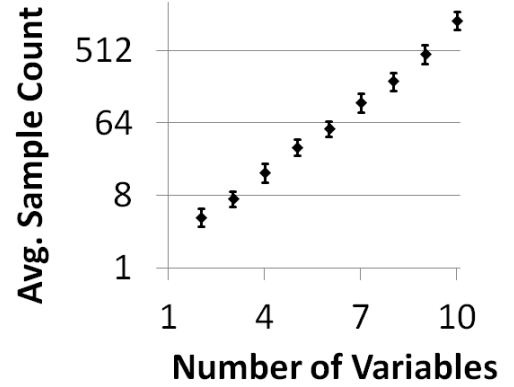


Figure 4: Sample complexity for CTBN inference.

for $n = 2, \dots, 9$. Each expression has exactly one truth assignment that satisfies it (all variables set to *True*). We count the number of samples generated until we have the first sample for which $D_m(0.2) = T$, making $P(D_m(0.2) = T) > 0$ and thus showing that the 3SAT instance is satisfiable. For each number of variables, we average the number of samples generated over 100 runs. The average sample counts along with confidence intervals for $\alpha = 0.01$ are plotted in Figure 4. The \log_2 scale on the y -axis shows that the algorithm is exponential in the length of the expression.

6 CONCLUSION

We have shown that exact and approximate inference in CTBNs is NP-hard, even when given the initial states. Thus, the difficulty of CTBN inference is found not only in Bayesian network inference for calculating the initial distribution, but also in accurately propagating the probabilities forward in time. Given the similar results with Bayesian networks, these results are not surprising. However, as with Bayesian networks, further research may reveal special cases of the CTBN, whether in their structures or their parameters, which admit polynomial-time algorithms for approximate or even exact inference.

References

- Chickering, D. M. (1996). Learning Bayesian networks is NP-complete. In *Learning from data* (pp. 121–130). Springer.
- Cohn, I. (2009). *Mean field variational approximations in continuous-time markov processes*. Unpublished doctoral dissertation, The Hebrew University.
- Cohn, I., El-Hay, T., Friedman, N., & Kupferman, R. (2009). Mean field variational approximation for

- continuous-time Bayesian networks. In *Proceedings of the twenty-fifth conference annual conference on uncertainty in artificial intelligence (uai-09)* (pp. 91–100). Corvallis, Oregon: AUAI Press.
- Cohn, I., El-Hay, T., Friedman, N., & Kupferman, R. (2010). Mean field variational approximation for continuous-time Bayesian networks. *The Journal of Machine Learning Research*, 11, 2745–2783.
- Cooper, G. (1990). The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial intelligence*, 42(2), 393–405.
- Dagum, P., & Luby, M. (1993). Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artificial intelligence*, 60(1), 141–153.
- El-Hay, T., Cohn, I., Friedman, N., & Kupferman, R. (2010). Continuous-time belief propagation. In *Proceedings of the 27th international conference on machine learning (icml)*.
- El-Hay, T., Friedman, N., & Kupferman, R. (2008). Gibbs sampling in factorized continuous-time Markov processes. In *Proceedings of the twenty-fourth conference annual conference on uncertainty in artificial intelligence (uai-08)* (pp. 169–178). Corvallis, Oregon: AUAI Press.
- Fan, Y. (2009). *Continuous time bayesian network approximate inference and social network applications*. Unpublished doctoral dissertation, University of California.
- Fan, Y., & Shelton, C. (2008). Sampling for approximate inference in continuous time Bayesian networks. In *Tenth international symposium on artificial intelligence and mathematics*.
- Fan, Y., & Shelton, C. (2009). Learning continuous-time social network dynamics. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence (uai-09)* (pp. 161–168).
- Fan, Y., Xu, J., & Shelton, C. R. (2010). Importance sampling for continuous time Bayesian networks. *The Journal of Machine Learning Research*, 99, 2115–2140.
- Gatti, E. (2011). *Graphical models for continuous time inference and decision making*. Unpublished doctoral dissertation, Università degli Studi di Milano-Bicocca.
- Gatti, E., Luciani, D., & Stella, F. (2011). A continuous time Bayesian network model for cardiogenic heart failure. *Flexible Services and Manufacturing Journal*, 1–20.
- Herbrich, R., Graepel, T., & Murphy, B. (2007). Structure from failure. In *Proceedings of the 2nd usenix workshop on tackling computer systems problems with machine learning techniques* (pp. 1–6).
- Koller, D., & Friedman, N. (2009). *Probabilistic graphical models: Principles and techniques*. MIT Press.
- Ng, B., Pfeffer, A., & Dearden, R. (2005). Continuous time particle filtering. In *International joint conference on artificial intelligence* (Vol. 19, p. 1360).
- Nodelman, U. (2007). *Continuous time Bayesian networks*. Unpublished doctoral dissertation, Stanford University, Stanford, California.
- Nodelman, U., & Horvitz, E. (2003). *Continuous time Bayesian networks for inferring users’ presence and activities with extensions for modeling and evaluation* (Tech. Rep.).
- Nodelman, U., Koller, D., & Shelton, C. (2005). Expectation propagation for continuous time Bayesian networks. In *Proceedings of the twenty-first conference annual conference on uncertainty in artificial intelligence (uai-05)* (pp. 431–440). Arlington, Virginia: AUAI Press.
- Qiao, S., Tang, C., Jin, H., Long, T., Dai, S., Ku, Y., & Chau, M. (2010). PutMode: prediction of uncertain trajectories in moving objects databases. *Applied Intelligence*, 33(3), 370–386.
- Rao, V., & Teh, Y. W. (2013). Fast MCMC sampling for Markov jump processes and extensions. *The Journal of Machine Learning Research*, 14(1), 3295–3320.
- Saria, S., Nodelman, U., & Koller, D. (2007). Reasoning at the right time granularity. In *Proceedings of the twenty-third conference annual conference on uncertainty in artificial intelligence (uai-07)*.
- Shi, D., Tang, X., & You, J. (2010). An intelligent system based on adaptive CTBN for uncertainty reasoning in sensor networks. *Intelligent Automation & Soft Computing*, 16(3), 337–351.
- Weiss, J. C., Natarajan, S., & Page, C. D. (2013). Learning when to reject an importance sample. In *Workshops at the twenty-seventh aaai conference on artificial intelligence*.
- Xu, J. (2010). *A continuous time bayesian network approach for intrusion detection*. Unpublished doctoral dissertation, University of California.
- Xu, J., & Shelton, C. (2008). Continuous Time Bayesian Networks for Host Level Network Intrusion Detection. In W. Daelemans, B. Goethals, & K. Morik (Eds.), *Machine learning and knowledge discovery in databases* (Vol. 5212, pp. 613–627). Springer Berlin / Heidelberg.
- Xu, J., & Shelton, C. R. (2010). Intrusion detection using continuous time Bayesian networks. *Journal of Artificial Intelligence Research*, 39(1), 745–774.

Model Regularization for Stable Sample Rollouts

Erik Talvitie

Department of Mathematics and Computer Science
Franklin & Marshall College
Lancaster, PA 17604

Abstract

When an imperfect model is used to generate sample rollouts, its errors tend to compound – a flawed sample is given as input to the model, which causes more errors, and so on. This presents a barrier to applying rollout-based planning algorithms to learned models. To address this issue, a training methodology called “hallucinated replay” is introduced, which adds samples from the model into the training data, thereby training the model to produce sensible predictions when its own samples are given as input. Capabilities and limitations of this approach are studied empirically. In several examples hallucinated replay allows effective planning with imperfect models while models trained using only real experience fail dramatically.

1 INTRODUCTION

Online Monte Carlo-based planning algorithms such as Sparse Sampling (Kearns et al. 2002), UCT (Kocsis & Szepesvári 2006), and POMCP (Silver & Veness 2010) are attractive in large problems because their computational complexity is independent of the size of the state space. This type of planning has been successful in many domains where a perfect model is available to the agent, but there is a fundamental barrier in the way of applying these methods to *learned* models. The core operation in Monte Carlo planning is sampling possible futures by composing the model’s one-step predictions (in a process called *rollout*). When an imperfect model is used, errors in one sampled observation cause more errors in the next sample, and so on until the rollout bears little or no resemblance to any plausible future, making it worthless for planning purposes.

To illustrate, consider the following simple planning problem, which will serve as a running example throughout the paper. In “Mini-Golf” (pictured in Figure 1a), the agent’s

observations are 20×3 binary images. The image shows a ball, a wall, and a pit. A cover slides back and forth over the pit. The agent selects between two actions: *no-op* and *hit*; once the *hit* action is selected, only the *no-op* action is available thereafter. When the ball is hit, it travels to the right, knocks the wall down (i.e., the wall disappears), bounces back to the left, hits the left edge, and bounces back to the right. If the cover is in the correct position, the ball reaches the right side, the episode ends, and the agent receives 1 reward. Otherwise, the ball falls into the pit which ends the episode with no reward.

Imagine that the agent learns a factored model over pixels. There is a separate component model responsible for predicting each pixel; the prediction for the full image is the product of the component models’ predictions. The component model at position $\langle x, y \rangle$ is given the pixel values in an $n \times m$ neighborhood centered at $\langle x, y \rangle$ for the previous two time-steps, as well as the most recent two actions.

If $n = 9$ and $m = 3$, there is enough information to make perfectly accurate predictions, but imagine that $n = 7$. This simulates the common situation that the minimum set of features necessary for perfect accuracy is either unknown or computationally impractical. The consequences of the limitation are illustrated in Figure 1b. The main problem is that a pixel immediately adjacent to the ball cannot tell if the ball is next to a wall, and therefore whether the ball will bounce back. As a result, the model for this pixel is uncertain whether the pixel will be black or white.

Figure 2 shows a rollout from a perfect model ($n = 9$) on the left and a rollout from an imperfect model ($n = 7$) in the center. When the model is very accurate, rollouts are plausible futures. When the model is imperfect, the model’s limitation eventually causes an error – a pixel next to the ball stochastically turns black, as if the ball were bouncing back. This creates a context for the nearby pixel models that is unlike any they have seen before. This causes new artifacts to appear in the next sample, and so on. The meaningful structure in the observations is quickly obliterated making it impossible to tell whether the ball will eventually reach the other side.



Figure 1: a) The Mini-Golf problem. b) The pixel marked with the ? cannot be predicted perfectly with a 7×3 neighborhood (shaded) because it cannot tell if the ball will bounce off of a wall.

This issue can be framed as a mismatch between the model’s training data and test data. Specifically, the model’s training data is drawn from real world experience but when it is actually used for planning (i.e., “tested”), the inputs to the model are samples generated from the model itself, which may be very different than real observations. In a sense, the model in this example has overfit to the real world! The approach taken in this paper to address this train/test mismatch is to mix samples from the model into the training data. The resulting method is called “hallucinated replay,” in analogy to “experience replay” (Lin 1992). On the right of Figure 2 is a rollout of a model trained using hallucinated replay. Note that it makes the same initial error (a spurious black pixel still appears), but it has learned that lone black pixels should turn white. In effect, it has learned to correct for its own sampling errors.

This paper has two main goals. The first is to highlight the dramatic impact of this mismatch between real inputs and sampled inputs on model-based reinforcement learning. In several examples seemingly innocuous model limitations cause catastrophic planning failures. The second is to empirically investigate hallucinated replay as an approach to addressing this issue. Most notably, hallucinated replay will be used to learn imperfect models that can nevertheless be used for effective planning. These results indicate that this method may be an important tool for model-based reinforcement learning in problems where one cannot expect to learn a perfect model.

2 ENVIRONMENTS AND MODELS

The development of hallucinated replay will focus on the setting of multi-dimensional, discrete, deterministic dynamical systems. Time proceeds in discrete steps. At each step t , the agent selects an action a_t from a finite set of actions \mathcal{A} . Given the action, the environment deterministically produces an observation vector \mathbf{o}_t . Let $\mathcal{O} = \mathcal{O}_1 \times \mathcal{O}_2 \times \dots \times \mathcal{O}_n$ be the observation space, where \mathcal{O}_i is the finite set of possible values for component i of the observation. Note that *not all* $\mathbf{o} \in \mathcal{O}$ will necessarily be reachable in the world. Let $\mathcal{O}_E \subseteq \mathcal{O}$ be the subset of observations that the environment can ever produce. For example, in Mini-Golf, \mathcal{O} would be the set of *all* 20×3 binary images, and \mathcal{O}_E would be the set of binary images with a ball, a pit, and either a wall or no wall.

Let \mathcal{H} be the set of all sequences $a_1 \mathbf{o}_1 \dots a_k \mathbf{o}_k$ for all

lengths k and let $\mathbf{o}_t = T(a_t, h_{t-1})$ where $h_{t-1} = a_1 \mathbf{o}_1 \dots a_{t-1} \mathbf{o}_{t-1}$ is the *history* at time $t-1$ and T is the *transition function* $T : \mathcal{A} \times \mathcal{H} \rightarrow \mathcal{O}_E$. Note that typically not all sequences in \mathcal{H} are reachable. Let $\mathcal{H}_E \subseteq \mathcal{H}$ be the set of histories that could be produced by taking some action sequence in the environment.

At each step the environment also produces a reward value r_t and, in the episodic case, a Boolean termination signal ω_t . For simplicity’s sake, these values are assumed to be contained in the observation vector \mathbf{o}_t so correctly predicting the next observation also involves correctly predicting the reward and whether the episode will terminate. Assume that if $\omega_t = \text{True}$ then $\omega_{t'} = \text{True}$ and $r_{t'} = 0$ for all $t' > t$. The goal of an agent in this environment is to maximize the expected discounted return $E[\sum_{t=1}^{\infty} \gamma^{t-1} r_t]$, where $\gamma \in [0, 1]$ is the discount factor and the expectation is over the agent’s behavior (the environment is deterministic).

Note that even though the environment is assumed to be deterministic, it may be too complex to be perfectly captured by a model. Thus models will, in general, be stochastic. Also note that because \mathcal{O}_E and \mathcal{H}_E are unknown *a priori*, models will be defined over the full spaces \mathcal{O} and \mathcal{H} instead. So, an imperfect model not only has an incorrect probability distribution over possible observations, but it may assign positive probability to *impossible* observations, as seen in the Mini-Golf example. Furthermore, during rollouts the model will have to make predictions given histories in $\mathcal{H} \setminus \mathcal{H}_E$, though no amount of experience in the environment will provide guidance about what those predictions should be.

In an abstract sense, a *model* M provides a conditional probability $T_M(\mathbf{o} \mid ha)$ for every observation vector $\mathbf{o} \in \mathcal{O}$, action $a \in \mathcal{A}$, and history sequence $h \in \mathcal{H}$. As assumed above, predicting the next observation also involves predicting the reward value and whether the episode will terminate. Note that the history h is an arbitrarily long and ever expanding sequence of actions and observations. Most practical models will rely upon some finite dimensional summary of history $c(ha)$, which is hereafter called the model’s *context*. For the sake of simplifying notation, let $c_t = c(h_{t-1} a_t)$. For instance, a Markov model’s context would include the most recent action and observation and a POMDP model’s context would include the belief state associated with the history.

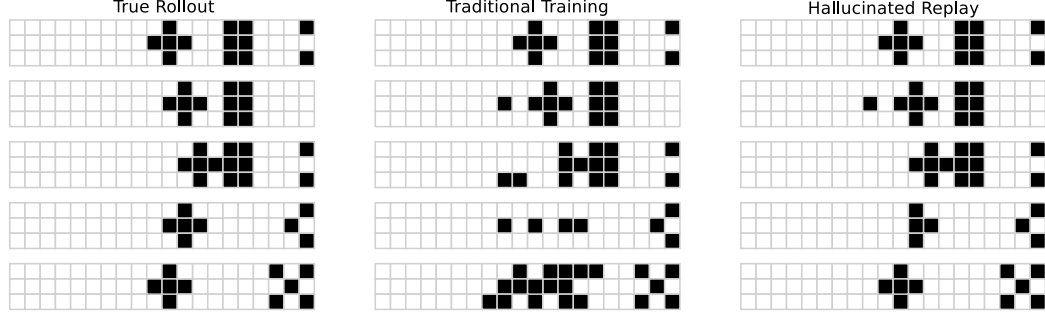


Figure 2: Rollouts using a perfect model (left), a model trained with only real experience (center) and a model trained with hallucinated replay (right). Hallucinated replay makes the model robust to its own errors.

3 HALLUCINATED REPLAY

Consider learning a model using a set of training trajectories $\{h^1, \dots, h^k\}$, where trajectory h^i is of length $|h^i|$. The training data can be seen as a set of training pairs for a supervised learning algorithm: $\{\langle \mathbf{c}_t^i, \mathbf{o}_t^i \rangle \mid i = 1 \dots k, t = 1 \dots |h^i|\}$, where the context \mathbf{c}_t^i is the input and the observation \mathbf{o}_t^i is the target output. Let \mathcal{M} be the space of possible models, or the *model class*. Most model learning methods aim to find the model $M^* \in \mathcal{M}$, which maximizes the log-likelihood:

$$\begin{aligned} M^* &= \operatorname{argmax}_{M \in \mathcal{M}} \sum_{i=1}^k \sum_{t=1}^{|h^i|} \log(T_M(\mathbf{o}_t^i \mid \mathbf{c}(h_{t-1}^i a_t^i))) \\ &= \operatorname{argmax}_{M \in \mathcal{M}} \sum_{i=1}^k \sum_{t=1}^{|h^i|} \log(T_M(\mathbf{o}_t^i \mid \mathbf{c}_t^i)) \end{aligned}$$

If a perfect model exists in \mathcal{M} , one for which $T_M(\mathbf{o}_t^i \mid \mathbf{c}_t^i) = 1$ for all i and t , then M^* is a perfect model. So in this case M^* is a sensible learning target for the purposes of model-based reinforcement learning.

If there is no perfect model in \mathcal{M} , M^* may not be a very good model for planning. Note that the log-likelihood only measures the accuracy of the model’s one-step predictions given contexts produced by the environment. However, as seen above, in order to prevent errors from compounding during planning, the model must make sensible predictions *in contexts the environment will never produce*. As will be demonstrated empirically, a model that accomplishes this may yield better planning performance than M^* , even if it incurs more prediction error (lower log-likelihood).

The high-level idea behind *hallucinated replay* is to train the model on contexts it will see during sample rollouts in addition to those from the world. Hallucinated replay augments the training data by randomly selecting a context \mathbf{c}_t^i from the training set, replacing it with a “hallucinated” context $\hat{\mathbf{c}}_t^i$ that contains sampled observations from the model, and finally adding the training pair $\langle \hat{\mathbf{c}}_t^i, \mathbf{o}_t^i \rangle$ to the training set. The model is trained with inputs that are generated

from its own predictions, but outputs from real experience.

The experiments below consider three concrete instantiations of this abstract idea, which are described below and illustrated in Figure 3. All three begin by randomly selecting a trajectory i and timestep t to select the “output” component of the new training pair.

Shallow Sample: The simplest form of hallucinated replay is to replace the observation at step $t - 1$ with a sample $\hat{\mathbf{o}}_{t-1}^i \sim T_M(\cdot \mid \mathbf{c}_{t-1}^i)$. Then the hallucinated context is $\hat{\mathbf{c}}_t^i = \mathbf{c}(h_{t-2}^i a_{t-1}^i \hat{\mathbf{o}}_{t-1}^i a_t^i)$. This approach essentially imagines that there might be errors in the first sample of a rollout (the sampled observation placed at the end of history) and trains the model to behave reasonably despite those errors (i.e., to predict the true next observation).

Deep Sample: A flaw in the above approach is that, even if a rollout recovers from an error the erroneous observation will still persist in history, possibly affecting predictions deeper in the rollout. A simple fix is to randomly select an observation in recent history (not just the most recent one) to replace with a sample. Specifically, randomly select an offset $j > 0$ and replace the observation at step $t - j$ with a sample $\hat{\mathbf{o}}_{t-j}^i \sim T_M(\cdot \mid \mathbf{c}_{t-j}^i)$ to generate the hallucinated context $\hat{\mathbf{c}}_t^i = \mathbf{c}(h_{t-j-1}^i a_{t-j}^i \hat{\mathbf{o}}_{t-j}^i a_{t-j+1}^i \mathbf{o}_{t-j+1}^i \dots a_{t-1}^i \mathbf{o}_{t-1}^i a_t^i)$. If the model is Markovian (that is, if its context only considers the most recent observation), then these two strategies are equivalent. In the experiments, the model is 2nd-order Markov, so j is either 1 or 2.

Deep Context: The above approaches have the drawback that they only introduce a single sampled observation into the context, whereas the contexts encountered by the model during a rollout will largely consist *entirely* of sampled observations. Even if the context only makes use of one observation, the above approaches sample that observation as if it is the first in the rollout, whereas the contexts encountered by the model will largely be sampled after a long rollout. To address this, randomly select a rollout length l and sample observations $\hat{\mathbf{o}}_j^i$ for $j = t - l \dots t - 1$. Then the hallucinated context is

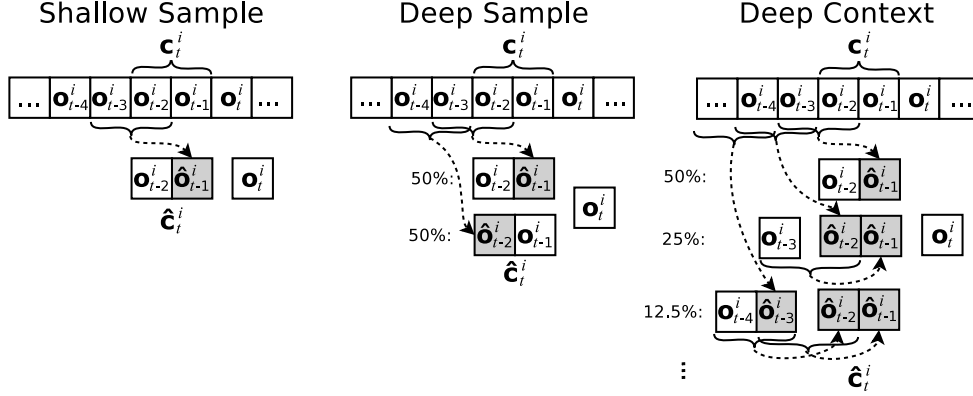


Figure 3: Methods for producing hallucinated contexts, illustrated with a 2nd-order Markov context. Samples are shaded; dashed arrows indicate the contexts which generated the samples. Actions are not shown.

$\hat{c}_t^i = c(h_{t-l-1}^i a_{t-l}^i \hat{o}_{t-l}^i \dots a_{t-1}^i \hat{o}_{t-1}^i a_t^i)$. In the experiments below, $l > 0$ is drawn with probability $\Pr(l) = \frac{1}{2^l}$. This biases the distribution toward short rollouts, since long rollouts are expected to produce very noisy observations.

3.1 IMPORTANCE OF DETERMINISM

Note that hallucinated replay as described can not sensibly be applied to stochastic environments. In a deterministic environment, the only reason \hat{c}_t^i can differ from c_t^i is that the model is imperfect. In a stochastic environment, even a perfect model could generate $\hat{c}_t^i \neq c_t^i$ by sampling a different stochastic outcome. In that case, training on the pair $\langle \hat{c}_t^i, o_t^i \rangle$ could harm the model by contradicting temporal dependencies the model may have correctly identified. One exception to this limitation is when the source of stochasticity is “measurement error,” noise that is uncorrelated over time and does not affect the evolution of the underlying state. In that case, hallucinated replay would re-sample the noise to no ill effect. This work focuses on the case of stochastic models of deterministic environments (assuming the environment may be too complex to model perfectly). This is, in itself, a large and interesting class of problems. The possibility of extending the idea to stochastic environments is briefly discussed in Section 5.2.

4 EXPERIMENTS

This section contains the main results of the paper, an empirical exploration of the properties of hallucinated replay under different modeling conditions. The experiments will be performed on variations of the Mini-Golf problem described in the introduction. One of the key features of this domain is that, though it is a simple planning problem (the agent needs only to select *hit* at the right moment), a long rollout is necessary to sample the eventual reward value and determine whether it is better to *hit* or *no-op*. The experiments employ the simple *1-ply Monte Carlo* algorithm:

at each step, for each available action a , perform 50 rollouts of length at most 80 that begin with a and uniformly randomly select actions thereafter. After the rollouts, execute the action that received the highest average return. In these experiments model learning and planning are both performed online – at each step, the planner uses the current model, which is then updated with the resulting training pair. All of the replay strategies train on 5 additional pairs from past experience per step.

The model of the pixels is factored, as described in the introduction. If a neighborhood contains pixels outside the boundaries of the image they are encoded with a special “out of bounds” color. In all the experiments data is shared across all positions, providing some degree of positional invariance in the predictions. To predict the reward and termination signals r_t and ω_t (which do not have positions on the screen) the context is the pixel values at all positions in o_t and o_{t-1} , as well as the actions a_t and a_{t-1} . During sampling the pixel values are sampled before reward and termination so the sampled image can be supplied as input.

In most of the experiments below the *Context Tree Weighting* (CTW) algorithm (Willems et al. 1995) is used to learn the component models, similar to the FAC-CTW algorithm (Veness et al. 2011), which was also applied in a model-based reinforcement learning setting. Very briefly, CTW maintains a Bayesian posterior over all prunings of a fixed decision tree. Amongst its attractive properties are computationally efficient updates, a lack of free parameters, and guaranteed zero asymptotic regret with respect to the maximum likelihood tree in the model class. In order to apply CTW, an ordering must be selected for the variables in the context tree. In all cases the actions and observations were ordered by recency. For the pixel models, neighboring pixels within a timestep were ordered by proximity to the pixel being predicted. For the reward and termination models, pixel values were in reverse column-major order (bottom-to-top, right-to-left).

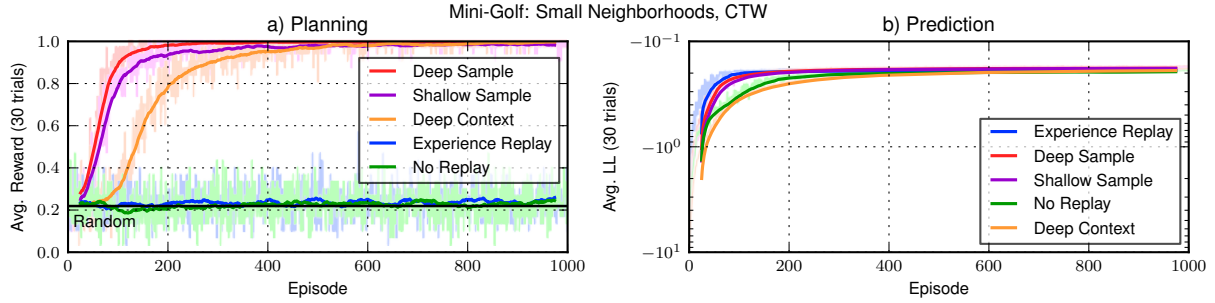


Figure 4: CTW results in Mini-Golf with small neighborhoods, averaged over 30 independent trials. Curves are smoothed using a sliding average of width 50. Unsmoothed data is shown in faint colors. The legends list the replay method in decreasing order of the value of the curve at episode 200 (in both cases, higher is better). Note the log scale in the prediction performance graph. These conventions are maintained in subsequent figures.

4.1 CORRECTING SAMPLE ERRORS

The first experiment considers the setting described in the introduction in which models are learned with 7×3 neighborhoods. Figure 4 shows the results of training with the three hallucinated replay variations proposed above and training with pure “Experience replay” (re-training on randomly selected training pairs from past experience without modifying the context). The results of training with only the agent’s experience (“No Replay”) are also provided as a baseline to show the effects of replay alone.

The most dramatic result can be seen in the planning performance (Figure 4a): the models trained only with experience from the environment (regardless of whether replay was used) result in behavior that is no better than an agent that chooses actions uniformly randomly (indicated by the solid black line). As can be seen in Figure 4b, the models trained using hallucinated replay achieve no better (and in one case much worse) prediction accuracy, as measured by average per-step log-likelihood over the training data (not counting replay). Nevertheless, they all result in near perfect planning performance. This matches the intuition described above, and indeed the rollouts in Figure 2 were generated using these learned models.

It is unsurprising that the Deep Sample replay strategy would outperform the Shallow Sample strategy – it was indeed important that sampled observations appear in both positions in the context. It is somewhat surprising that Deep Sample outperformed Deep Context as well. The hypothesis that it would be important for the model to be trained on contexts sampled far into a rollout is not supported by these results. It may be that the noisy samples obtained from long rollouts, especially early in training, were misleading enough to harm performance. It is also possible that this effect is an artifact of this example. Nevertheless, these qualitative findings were consistent across the variations explored below so, to reduce clutter, only Deep Sample results are reported in the subsequent experiments.

4.2 FULL NEIGHBORHOODS

The previous experiment is an example of hallucinated replay making meaningful planning possible when a perfect model is not contained within the model class. This is the most common case in problems of genuine interest. Nevertheless, it is relevant to ask what effect it has when a perfect model *is* in the class.

Figures 5a and 5b show the results in the Mini-Golf problem when the models used 9×3 neighborhoods (sufficient to make accurate predictions). Both the Experience Replay and No Replay models are able to support planning, and experience replay clearly improves sample efficiency. The Hallucinated Replay model has consistently lower prediction accuracy and lags behind the Experience Replay model in planning performance, though it eventually catches up. Noting that the hallucinated contexts in the very beginning of training are likely to be essentially meaningless, Figure 5 also shows the results of performing experience replay for the first 50 episodes, and hallucinated replay thereafter. This model’s planning performance is nearly identical to that of the Experience Replay model. Once the model has had some time to learn, the sampled observations become more and more like real observations, so this similarity is not surprising.

Note that though a perfect model is in the class, the intermediate models during learning are imperfect and their sampling errors do impact planning performance. It is interesting that hallucinated replay does not aid planning performance by making the model more robust to these errors. One hypothesis might be that if the rollouts were much longer, errors would be more likely to affect planning, and hallucinated replay might have more of an impact.

Figures 5c and 5d show the results of a variation on Mini-Golf that is the same in every respect, except that there are 6 walls to knock down rather than 1. Thus rollouts must be *much* longer for successful planning. First note that the

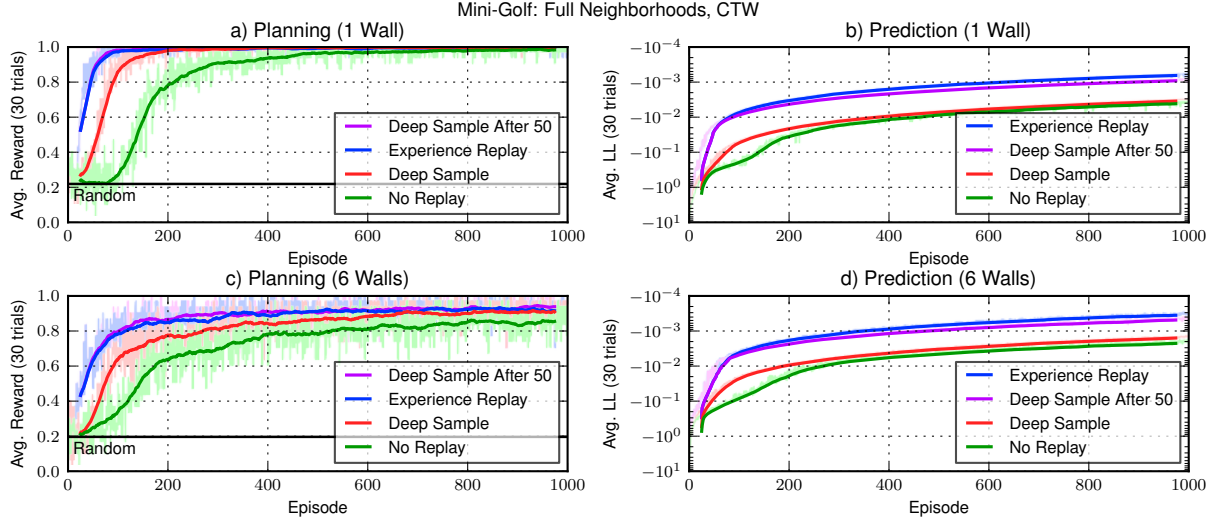


Figure 5: CTW results in Mini-Golf with full neighborhoods, both with 1 wall and with 6 walls.

hypothesis that longer rollouts will cause more sampling errors is borne out. For a comparable level of prediction accuracy, the models achieve substantially better planning performance in the 1 wall version than the 6 wall version. Still, hallucinated replay was not able to mitigate the impact of the model errors.

The errors produced by these “models-in-training” tend to be “salt-and-pepper” noise caused by pixel models that all assign small but positive probability to the incorrect outcome. This is a relatively unstructured type of noise in comparison to that seen in the first experiment. It may simply be that more data is required to learn to correct for the noise than to simply improve accuracy. Or, since objects in this problem consist of only a few pixels, this type of noise may be too destructive to be compensated for.

4.3 A DIFFERENT MODEL CLASS

In order to validate that the benefits of hallucinated replay are robust to the choice of model class, Figure 6 shows the results of training feed forward neural networks in the Mini-Golf domain. The same model architecture was used except here the pixel, reward, and termination models are neural networks with 10 logistic hidden units and a logistic output layer, trained using standard backpropagation (Rumelhart et al. 1986). Several values of the learning rate α were tested and the results using the best value of α for each method are presented. Figures 6a and 6b show the planning and prediction performance of the neural network models when given 7×3 neighborhoods. The results are very similar to those using CTW models. Figures 6c and 6d show the results using 9×3 neighborhoods. Here, unlike with CTW, the hallucinated replay training does not seem to cause a significant loss in performance. The neural

network is likely less sensitive to the initially uninformative hallucinated samples because it is easily able to ignore irrelevant features.

Notably, waiting to start hallucinated replay *hurts* performance rather than helps. The reason for this can be seen in the prediction performance graphs: a large drop in prediction performance is observed when hallucinated replay begins – this is because the hidden units must adjust to a sudden change in the distribution of input vectors. This indicates that the compatibility of a model learning method with hallucinated replay may depend on its ability to gracefully handle the non-stationarity it creates in the training data. No-regret algorithms like CTW may be particularly well-suited for that reason.

Neural network models also provide an opportunity to examine another source of model error. Figure 7 shows the results (planning only) of using neural network models with too few hidden nodes (but with full neighborhoods). In Figure 7a, the networks have 5 hidden nodes. The models trained with experience replay are able to learn quickly, but level off at sub-optimal behavior. Training with hallucinated replay is slower, but planning performance ultimately surpasses that of the experience replay model. In Figure 7b, the average score of the last 100 episodes (out of 5000) is shown for various numbers of hidden nodes. The best learning rate for each method and each number of hidden nodes is used. With a small number of hidden nodes, the model is unable to make good predictions or correct for errors. With a large number, an accurate model can be learned and hallucinated replay is unnecessary. For intermediate values hallucinated replay is able to compensate somewhat for the model’s representational deficiency.

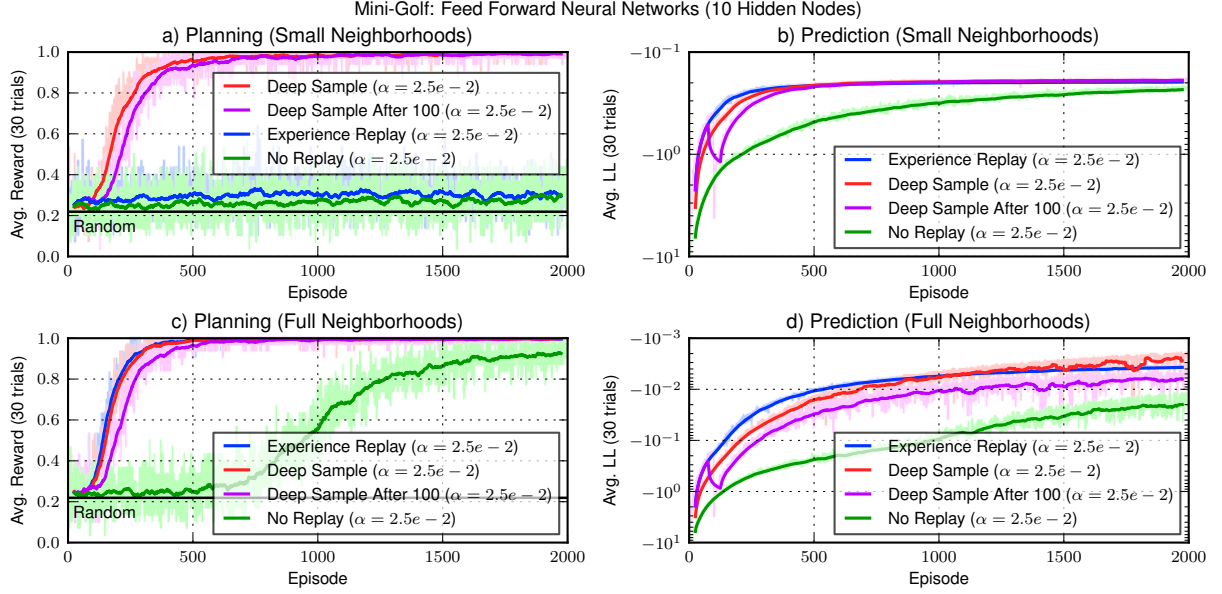


Figure 6: Neural network results in Mini-Golf with 10 hidden nodes.

4.4 ADDITIONAL EXAMPLES

Finally, as additional examples of model impairments that hallucinated replay can help to overcome, consider two variations on the Mini-Golf domain.

4.4.1 DECEPTIVELY INFORMATIVE FEATURES

In the first variation, a score display is added. The observations are now 20×6 images. The bottom three rows of the image are blank for the most part, but at the end of an episode the 3×3 area in the bottom right displays a digit (“0” or a “1”) to show the reward received.

As can be seen in Figure 8a, when the model is trained using only real experience this seemingly innocuous addition causes severe planning problems, even though the pixel models are given sufficient contexts to perfectly model the dynamics of the game. The problem arises because the neighborhood size is *not* sufficient to model the score display. When trained only on real experience, the reward and termination models rightly learn that the score display is a reliable predictor. However, during rollouts, those pixels no longer behave as they do in the world, and the reward and termination models make erratic predictions.

In contrast, the models trained with hallucinated replay learn that the score display pixels cannot be relied upon and focus instead on other relevant contextual information (such as the ball’s position). This allows the models to provide meaningful predictions that result in good planning performance. Note again that hallucinated replay has not fixed the errors. The hallucinated replay models are no bet-

ter at predicting what the score display will do. They are able to make good predictions about the reward and termination signals *despite* sample errors in the score display.

4.4.2 COMPLICATED, IRRELEVANT FEATURES

In the second variation, an irrelevant but hard-to-predict component is added to the image. Specifically, the images are 21×3 where the last column displays a set of “blinking lights.” Only one pixel in the last column is white on any given step, but the pattern of which pixel is white is 3rd-order Markov. Since the model is 2nd-order Markov, it cannot reliably predict the light pattern. That said, the lights have no impact on the rest of the dynamics and the models have sufficient context to make perfect predictions for every other pixel. Nevertheless, as can be seen in Figure 8b, this addition once again causes the model trained only on real experience to fail when used for planning.

In this case the problem is that rollout samples of the blinking lights will not resemble those seen in the world. In the world, only one light can be on at a time but in samples lights stochastically turn on and off, causing novel configurations. The pixel models neighboring these unfamiliar configurations have higher uncertainty as a result, and the now familiar error compounding process ensues, corrupting the important parts of the sampled image as well as the unimportant parts. The models trained using hallucinated replay, in contrast, learn to make predictions given the distribution of light configurations that will actually be encountered during a rollout.

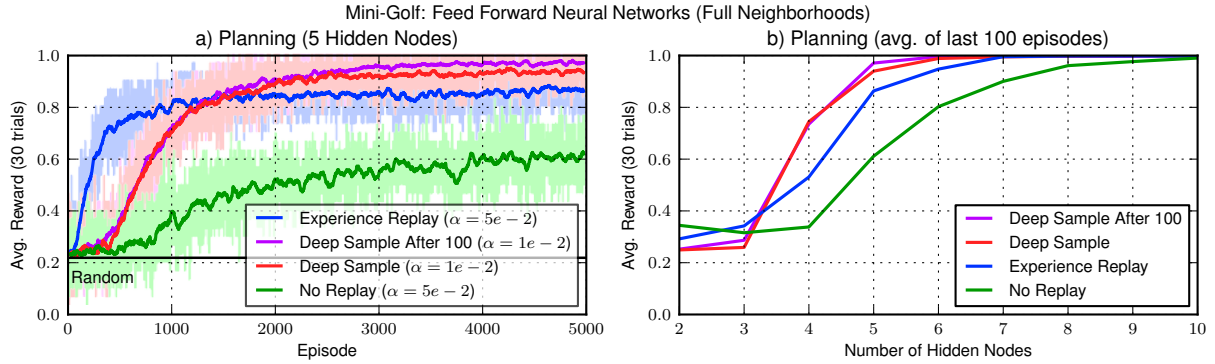


Figure 7: Neural network results in Mini-Golf with full neighborhoods but too few hidden nodes.

5 DISCUSSION

The results have shown that even mildly limited factored models that can accurately predict nearly every aspect of the environment’s dynamics can dramatically fail when used in combination with Monte-Carlo planning. Since factored models and Monte-Carlo planning are both important, successful tools in problems with large state/observation spaces, resolving this issue may be a key step toward scaling up model-based reinforcement learning. Overall, hallucinated replay was effective in its purpose: to train models to be more robust to their own sampling errors during rollouts.

When a perfect model was contained within the class (a rare occurrence in problems of genuine interest), hallucinated replay was comparable to (but slightly less effective than) pure experience replay. Though in this case the models are imperfect during training, hallucinated replay did not seem to be effective at compensating for the type of wide-spread, unstructured noise that resulted from transient parameter settings (as opposed to the more systematic errors resulting from class limitations in other examples).

It was also observed that when the model class is *extremely* limited (e.g. the neural network models with only 2 hidden nodes), hallucinated replay may be ineffective at compensating for errors because the model simply cannot learn to make meaningful predictions at all. Of course, in this case the model is ineffective no matter how it is trained. It seems that hallucinated replay is most effective when the model can capture *some but not all* of the environment’s dynamics. When a model is performing poorly hallucinated replay may be able to magnify the impact of improving the model’s expressive power (as was seen when the neural networks were given more hidden nodes).

5.1 RELATED WORK

Other authors have noted that the most accurate model in a limited class may not be the best model for planning. Sorg,

Singh & Lewis (2010) argue that when the model or planner are limited, reward functions other than the true reward may lead to better planning performance. Sorg, Lewis & Singh (2010) use policy gradient to learn a reward function for use with online planning algorithms. Joseph et al. (2013) make a similar observation regarding the dynamics model and use policy gradient to learn model parameters that lead to good planning, rather than low prediction error (demonstrating that the two do not always coincide). The results presented here add to the evidence that simply training to maximize one-step prediction accuracy may not yield the best planning results.

Siddiqi et al. (2007) addressed the problem of learning stable dynamics models of linear dynamical systems. The key idea was to project the learned dynamics matrix into the constrained space of matrices with spectral radius of at most 1, and which therefore remain bounded even when multiplied with themselves arbitrarily many times. Though certainly conceptually related (their work has the same goal of learning a model whose predictions are sensible when given its own output as input), their specific approach is unlikely to apply to the setting considered here, as the concept of “stability” is not so easily quantified.

There is a long history in the supervised learning setting of adding noise to training data (or otherwise intentionally corrupting it) to obtain regularization and generalization benefits (see e.g. Matsuoka 1992, Burges & Schölkopf 1997, Maaten et al. 2013). There the issue is not typically prediction composition but more broadly the existence of inputs that the training set may not adequately cover. Without access to the true distribution over inputs, noise is typically generated via some computationally or analytically convenient distribution (e.g. Gaussian) or via a distribution that incorporates prior knowledge about the problem. In contrast, we have access to the model which generates its own inputs, and can thus sample corrupting noise from a learned, but more directly relevant distribution.

Ross & Bagnell (2012) observed a similar train/test mis-

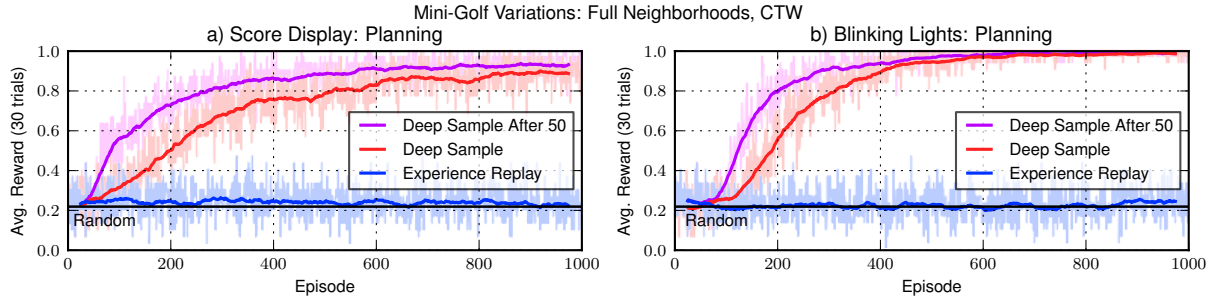


Figure 8: CTW results in two variations of Mini-Golf (described in text).

match in the model-based reinforcement learning setting when the model is trained on a fixed batch of data. In that case, the plan generated using the model may visit states that were underrepresented in training, resulting in poor performance. Their algorithm DAGger mixes experience obtained using model-generated plans into the training data as the model learns, thereby closing the gap between training and test distributions. Roughly speaking, they prove that their approach achieves good planning performance when a model in the class has low prediction error. In the examples considered here *no* model in the class has low prediction error, but near perfect planning performance is nevertheless possible.

5.2 FUTURE DIRECTIONS

The results indicate that, in addition to one-step accuracy, a model’s robustness to its own errors should be a key concern for model based reinforcement learning. They also indicate that training a model on its own outputs is a promising approach to addressing this concern. These observations open the door to a number of interesting questions and issues for further study.

As discussed in Section 3, hallucinated replay is not immediately applicable in stochastic domains. While a lot of perceived stochasticity can be attributed to unmodeled (but deterministic) complexity, it would nevertheless be valuable to explore whether this approach can be applied to inherently stochastic dynamics. The main challenge in this case is forcing the model’s hallucinated data to choose the same stochastic outcome that was observed in the real data. It may be that this could be accomplished by learning reverse correlations from real world outcomes to hallucinated contexts and applying rejection sampling when generating hallucinated data.

The hallucinated replay algorithm presented in this paper is simple and offers no theoretical guarantees. It may, however, be possible to incorporate the basic idea of hallucinated replay into more principled algorithms that admit more formal analysis. For instance, it may be possible to extend the DAGger algorithm given by Ross & Bagnell

(2012) using a form of hallucinated replay. This would allow the analysis to take into account the learned model’s robustness (or lack thereof) to its own errors. It would also be valuable, if possible, to develop algorithms that have the benefits of hallucinated replay but are able to promise that the hallucinated data will not harm asymptotic performance if an accurate model is in the class.

Perhaps most important would be to develop a more formal understanding of the relationships between accuracy, robustness, and planning performance. In sufficiently interesting environments, model errors will be inevitable. Successfully planning despite those errors is key to applying model based reinforcement learning to larger, more complex problems. A characterization of the properties beyond simple accuracy that make a model good for planning and a more nuanced understanding of which types of model error are acceptable and which are catastrophic could yield improved forms of hallucinated replay, or other model learning approaches that have good planning performance as an explicit goal rather than an implicit effect of accuracy.

6 CONCLUSIONS

Hallucinated replay trains a model using inputs drawn from its own predictions, making it more robust when its predictions are composed. This training methodology was empirically shown to substantially improve planning performance compared to using only real experience. The results suggest that hallucinated replay is most effective when combined with model-learning methods that gracefully handle non-stationarity and when the model class’ limitations allow it to capture some but not all of the environment’s dynamics.

Acknowledgements

Thank you to Michael Bowling, Marc Bellemare, and Joel Veness for discussions that helped shape this work. Thanks also to Joel Veness for his freely available FAC-CTW implementation (<http://jveness.info/software/mc-aixi-src-1.0.zip>).

References

- Burges, C. J. & Schölkopf, B. (1997), Improving the accuracy and speed of support vector machines, in 'Advances in Neural Information Processing Systems (NIPS)', pp. 375–381.
- Joseph, J., Geramifard, A., Roberts, J. W., How, J. P. & Roy, N. (2013), Reinforcement learning with misspecified model classes, in '2013 IEEE International Conference on Robotics and Automation (ICRA)', pp. 939–946.
- Kearns, M., Mansour, Y. & Ng, A. Y. (2002), 'A sparse sampling algorithm for near-optimal planning in large markov decision processes', *Machine Learning* **49**(2-3), 193–208.
- Kocsis, L. & Szepesvári, C. (2006), Bandit based monte-carlo planning, in 'Proceedings of the 17th European Conference on Machine Learning (ECML)', pp. 282–293.
- Lin, L.-J. (1992), 'Self-improving reactive agents based on reinforcement learning, planning and teaching', *Machine learning* **8**(3-4), 293–321.
- Maaten, L., Chen, M., Tyree, S. & Weinberger, K. Q. (2013), Learning with marginalized corrupted features, in 'Proceedings of the 30th International Conference on Machine Learning (ICML-13)', pp. 410–418.
- Matsuoka, K. (1992), 'Noise injection into inputs in back-propagation learning', *IEEE Transactions on Systems, Man and Cybernetics* **22**(3), 436–440.
- Ross, S. & Bagnell, D. (2012), Agnostic system identification for model-based reinforcement learning, in 'Proceedings of the 29th International Conference on Machine Learning (ICML-12)', pp. 1703–1710.
- Rumelhart, D. E., Hinton, G. E. & Williams, R. J. (1986), 'Learning representations by back-propagating errors', *Nature* **323**(9), 533–536.
- Siddiqi, S. M., Boots, B. & Gordon, G. J. (2007), A constraint generation approach to learning stable linear dynamical systems, in 'Advances in Neural Information Processing Systems (NIPS)', pp. 1329–1336.
- Silver, D. & Veness, J. (2010), Monte-carlo planning in large pomdps, in 'Advances in Neural Information Processing Systems (NIPS)', pp. 2164–2172.
- Sorg, J., Lewis, R. L. & Singh, S. (2010), Reward design via online gradient ascent, in 'Advances in Neural Information Processing Systems (NIPS)', pp. 2190–2198.
- Sorg, J., Singh, S. & Lewis, R. L. (2010), Internal rewards mitigate agent boundedness, in 'Proceedings of the 27th International Conference on Machine Learning (ICML-10)', pp. 1007–1014.
- Veness, J., Ng, K. S., Hutter, M., Uther, W. T. B. & Silver, D. (2011), 'A Monte-Carlo AIXI Approximation', *Journal of Artificial Intelligence Research* **40**, 95–142.
- Willems, F. M., Shtarkov, Y. M. & Tjalkens, T. J. (1995), 'The context tree weighting method: Basic properties', *IEEE Transactions on Information Theory* **41**, 653–664.

HELM: Highly Efficient Learning of Mixed copula networks

Yaniv Tenzer

Department of Statistics
The Hebrew University
yaniv.tenzer@gmail.com

Gal Elidan

Department of Statistics
The Hebrew University
galel@huji.ac.il

Abstract

Learning the structure of probabilistic graphical models for complex real-valued domains is a formidable computational challenge. This inevitably leads to significant modelling compromises such as discretization or the use of a simplistic Gaussian representation. In this work we address the challenge of *efficiently* learning truly expressive copula-based networks that facilitate a mix of varied copula families within the *same* model. Our approach is based on a simple but powerful bivariate building block that is used to highly efficiently perform local model selection, thus bypassing much of computational burden involved in structure learning. We show how this building block can be used to learn general networks and demonstrate its effectiveness on varied and sizeable real-life domains. Importantly, favorable identification and generalization performance come with dramatic runtime improvements. Indeed, the benefits are such that they allow us to tackle domains that are prohibitive when using a standard learning approaches.

1 INTRODUCTION

Probabilistic graphical models [Pearl, 1988] in general and Bayesian networks (BNs) in particular, have become popular as a flexible and intuitive framework for modeling multivariate densities, a central goal of the data sciences. At the heart of the formalism is a combination of a qualitative graph structure that encodes the regularities (independencies) of the domain and quantitative local conditional densities of a variable given its parents in the graph. The result is a decomposable model that facilitates relatively efficient inference and estimation. Unfortunately, learning the structure of such models remains a formidable challenge, particularly when dealing with real-valued domains that are non-Gaussian. The computational bottleneck lies in the need to

assess the merit of many candidate structures, each requiring potentially costly maximum likelihood evaluation.

The situation is further compounded in realistic domains where we also want to allow for the combination of different local representations within the same model. Specifically, such a scenario requires that we perform non-trivial local model selection *within* an already challenging structure learning procedure. In practice, with as few as tens of variables, learning any real-valued graphical model beyond the simple linear Gaussian BN can be computationally impractical. At the same time, it is clear that, for many domains, the Gaussian representation is too restrictive. Our goal in this work is to overcome this barrier and to *efficiently* learn the structure of expressive networks that do not only go beyond the Gaussian, but that also allow for a mix of varied local representations.

In the search for expressive representations, several recent works use copulas as a building block within the framework of graphical models [Kirshner, 2007, Elidan, 2010, Wilson and Ghahramani, 2010]. Briefly, copulas [Joe, 1997, Nelsen, 2007] flexibly capture distributions of few dimensions: easy to estimate univariate marginals are joined together using a copula function that focuses solely on the dependence pattern of the joint distribution. Appealingly, regardless of the dependency pattern, any univariate representation can be combined with any copula. In all of the above works, the resulting copula graphical model proved quite effective at capturing complex high-dimensional domains, far surpassing the Gaussian representation.

Recently, Elidan [2012] proposed a structure learning method that is tailored to the so called copula network representation, and that is essentially as efficient as learning a simple linear Gaussian BN. However, an important drawback of the approach is that it constrains all local copulas in the model to be of the same type. Tenzer and Elidan [2013] offer a slight improvement but their method is inherently limited to few (2-3) of *specific* local representations and to tree-structured networks. Clearly, to take advantage of the plethora of dependency patterns captured by different copula families, we would like to have greater flexibility.

Unfortunately, selection of the right copula family, or dependence pattern, can be hard even for just two random variables. Typical approaches (e.g., [Huard et al., 2006, Fermanian, 2005, Hering and Hofert, 2010, Justel et al., 1997, Genest and Rivest, 1993]) require costly computations such as maximum likelihood estimation, Bayesian integration, simulation, etc. (see Section 3 for details). While such methods can be used to perform model selection for a distribution with few variables, they are impractical when faced with a large number of local model selection tasks that underlie global structure learning. In this work we introduce HELM: a method for **H**ighly **E**fficient **L**earning of **M**ixed copula networks.

Intuitively, for the task of model selection, the maximum likelihood density defined by a particular copula family is in fact a nuisance parameter, and we are only interested in detecting the dependency pattern of the copula. Further, since most copulas have a functional form with few parameters (or even just one), identifying between different copulas only requires a crude view of the distribution. Building on this intuition, we build a copula-to-multinomial mapping that is independent of a particular domain. Then, when faced with the model selection task given training samples, we use a comparison of the empirical multinomial signature to the precomputed mapping in order to choose the most promising copula family. Appealingly, for the building block task of choosing a copula family for two variables, our approach is effective, highly efficient, and comes with finite sample guarantees.

With this model selection building block in hand, we are still faced with the task of learning the *global* structure of the model, which in turn requires costly maximum likelihood computations. Fortunately, the same mechanism we use for selection suggests a highly efficient and effective proxy to the exact computation, when learning tree networks. Further, the method also gives rise to a natural heuristic generalization that allows us to highly efficiently learn networks with a general structure.

We demonstrate the benefit of our HELM approach for learning expressive networks that combine a varied set of copulas for several sizeable real-life datasets. Specifically, we show that our procedure is not only accurate in terms of identifying the best copula family, but also leads to learned probabilistic graphical models that generalize well. Importantly, this favorable performance comes with dramatic runtime speedups that facilitate learning of models in domains where maximum likelihood structure learning is computationally impractical.

2 BACKGROUND

In this section we briefly describe copulas, their relationship to Spearman’s ρ_s measure of association, and the copula network construction.

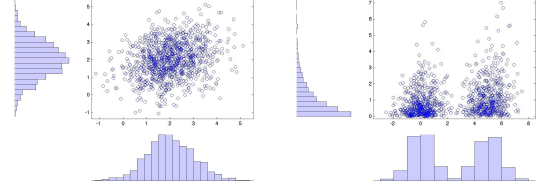


Figure 1: Samples from the bivariate Gaussian copula with $\rho = 0.25$. (left) with Gaussian marginals; (right) with a mixture of Gaussian and Gamma marginals.

Copulas

A copula function joins univariate marginals into a joint real-valued multivariate distributions. Formally, let U_1, \dots, U_n be random variables marginally uniformly distributed on $[0, 1]$. A copula function $C : [0, 1]^n \rightarrow [0, 1]$ is a joint distribution $C_\theta(u_1, \dots, u_n) = P(U_1 \leq u_1, \dots, U_n \leq u_n)$, where θ are the parameters of the copula distribution function.

Now let $\mathcal{X} = \{X_1, \dots, X_n\}$ be an arbitrary set of real-valued random variables. Sklar [1959] states that for *any* CDF $F_{\mathcal{X}}(\mathbf{x})$, there exists a copula C such that

$$F_{\mathcal{X}}(\mathbf{x}) = C(F_1(x_1), \dots, F_n(x_n)).$$

When $F_i(x_i)$ are continuous, C is uniquely defined.

The constructive converse is of particular interest from a modeling perspective: Since $F_i(X_i) \sim U([0, 1])$, *any* copula function taking *any* marginals $\{F_i(X_i)\}$ defines a valid joint cumulative distribution with marginals $\{F_i(X_i)\}$. Thus, copulas are “distribution generating” functions that allow us to separate the choice of the univariate marginals and that of the dependence.

To derive the joint *density* $f(\mathbf{x}) = \frac{\partial^n F(\mathbf{x})}{\partial x_1 \dots \partial x_n}$ from the copula construction, assuming F has n -order partial derivatives (true almost everywhere when F is continuous), and using the chain rule, we have

$$\begin{aligned} f(\mathbf{x}) &= \frac{\partial^n C(F_1(x_1), \dots, F_n(x_n))}{\partial F_1(x_1) \dots \partial F_n(x_n)} \prod_i f_i(x_i) \\ &\equiv c(F_1(x_1), \dots, F_n(x_n)) \prod_i f_i(x_i), \end{aligned}$$

where we use $c(F_1(x_1), \dots, F_n(x_n))$ to denote the *copula density function*.

Example 2.1.: The extremely popular Gaussian copula is defined as

$$C_{\Sigma}(\{U_i\}) = \Phi_{\Sigma}(\Phi^{-1}(U_1), \dots, \Phi^{-1}(U_N)), \quad (1)$$

where Φ is the standard Gaussian, and Φ_{Σ} is a zero mean Gaussian with correlation matrix Σ .

Copulas and Spearman's Rho

Copulas are intimately connected to many dependence concepts such as Spearman's ρ_s measure of association

$$\rho_s(X_1, X_2) = \frac{\text{cov}(F_{X_1}, F_{X_2})}{\text{STD}(F_{X_1})\text{STD}(F_{X_2})},$$

which is simply Pearson's correlation applied to the cumulative distributions of X_1 and X_2 . For the copula associated with the joint $F_{X_1, X_2}(x_1, x_2)$, we have

$$\rho_s(X_1, X_2) = \rho_s(C) \equiv 12 \int \int C(u, v) du dv - 3.$$

Thus, Spearman's ρ_s is monotonic in the copula cumulative distribution function associated with the joint distribution of X_1 and X_2 . See [Nelsen, 2007, Joe, 1997] for an in-depth exploration of the framework of copulas and its relationship to dependence measures.

Copula Networks

Similarly to a standard Bayesian network [Pearl, 1988], a copula network uses a directed acyclic graph \mathcal{G} to encode the independencies $I(\mathcal{G}) = \{(X_i \perp \text{NonDesc}_i \mid \mathbf{Pa}_i)\}$, where \mathbf{Pa}_i are the parents of X_i in \mathcal{G} , and NonDesc_i are its non-descendants. $I(\mathcal{G})$ implies a decomposition of the joint density into a product of local conditional densities of each variable given its parents: $f_{\mathcal{X}}(X_1, \dots, X_n) = \prod_i f_i(X_i \mid \mathbf{Pa}_i)$.

In copula networks, the local densities are defined via the copula ratio

$$f_i(X_i \mid \mathbf{Pa}_i) = \frac{c_{\theta}(F_i(X_i), \{F_j(X_j)\}_{j \in \mathbf{Pa}_i})}{c_{\theta}(\{F_j(X_j)\}_{j \in \mathbf{Pa}_i})} f_i(X_i). \quad (2)$$

Appealingly, for copulas the denominator can be easily computed from the numerator without the need for integration. Thus, the representation relies solely on the estimation of joint copulas. See [Elidan, 2010] for more details on the construction and its merits.

3 RELATED WORKS

Broadly speaking, methods for performing copula model selection can be split into three groups. Most commonly, model selection is carried out via (penalized) maximum likelihood estimation, which can be costly due to the need to evaluate the maximum likelihood parameters. In fact, as will be demonstrated in Section 7, even when the maximum likelihood parameters have a simple closed form, the actual computation of the maximum likelihood value can be time consuming in the context of structure learning, where this task is repeated numerous times. A second group of works relies on a measure of deviation between the copula, or some of its statistical properties, from the empirical estimators. Genest and Rivest [1993], for example, use the deviation of Kendall's τ estimates from the population values to select between Archimedean copulas. Unfortunately,

for most copulas, characterizing the Kendall distribution requires simulation and can be computationally demanding [Hering and Hofert, 2010]. Another example first employs Rosenblatt's transformation, followed by a deviation measurement relative to the uniform distribution [Justel et al., 1997], a process that can also be computationally intensive.

Fermanian [2005] suggests an alternative in the form of a goodness-of-fit test that is based on kernel density estimation. This, however, still requires tedious numerical integration. Finally, Huard et al. [2006] presents a Bayesian approach that, like our method, is quite generic as it avoids estimation of the maximum likelihood parameters. Posterior computations, however, still require costly integration over the support of Kendall's τ values. In contrast to all of these works, our HELM method uses extremely simple statistics that are easily computed for any copula. As demonstrated in Section 7, this leads to effective performance while offering dramatic speedups.

4 EXPRESSIVE TREE NETWORKS

Our goal is to efficiently learn the structure of copula-based probabilistic graphical models while allowing for different copula families within the same model. We start by considering in this section the building block task of performing selection for bivariate copulas, and show how this building block can be used to learn tree structured networks. Then, after deriving in Section 5 finite sample guarantees for the bivariate case, in Section 6 we propose an extension for learning general networks.

4.1 MULTINOMIAL-BASED SELECTION

Recall that, intuitively, for the purpose of choosing a particular dependence pattern, the distribution is a nuisance parameter and it may be possible to forgo precise estimation. As an example, Figure 2(top) shows the grid frequency of samples from the bivariate Clayton and Gumbel copulas with $\rho_s = 0.5$. The greater emphasis of the Clayton copula on the lower tail is evident as is the converse for the Gumbel copula. Thus, it is possible to choose between the copulas based on simple statistics. Motivated by this example, our selection procedure involves three steps:

1. Precompute a *multinomial signature* for each copula family under consideration.
2. Given a set of training instances, compute an empirical multinomial signature.
3. Choose the copula whose multinomial signature is closest (in some sense) to the empirical one.

We now briefly describe the details involved in each of these three stages.

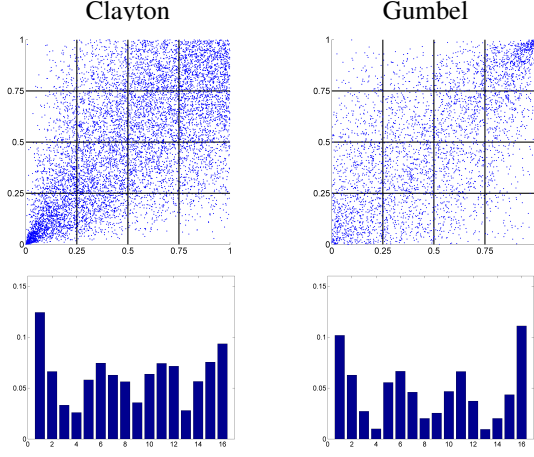


Figure 2: An example of the multinomial mapping for $\rho_s = 0.5$. **(top)** shows the distribution for the Clayton and Gumbel families, overlaid with a 4×4 grid. **(bottom)** shows the corresponding *multinomial signatures* defined by this grid for the two copula families.

Precomputing The Multinomial Signatures

We start by recalling the monotonic relationship, within a copula family, between Spearman’s ρ_s and the copula parameter. This allows us to use the notation C_{ρ_s} to refer to a particular instance of the copula family $C_{\theta(\rho_s)}$. To define the copula-to-multinomial mapping, for a copula family C_{ρ_s} , we partition the unit cube into N partitions $\{A_1, \dots, A_N\}$ and define a multinomial random variable X via

$$P_{C_{\rho_s}}(X = i) = \int_{A_i} c_{\rho_s}(u, v) du dv,$$

where c_{ρ_s} is the corresponding copula density. This mapping defines an N -valued multinomial representation of the copula which we denote by $\pi(c_{\rho_s})$. In principle, computing $P(X = i)$ requires integration. However, for copulas the cumulative distribution function is explicit and, if each A_i is chosen as a rectangular region, then $P(X = i)$ can be readily computed.

For simplicity, we use a generic partition into equal $K \times K$ squares as illustrated in Figure 2(top) for $K = 4$. Finally, since we map the copula to a crude coarsening as it is, in practice we compute the above only for $\rho_s \in \{-1, -0.95, \dots, 0.95, 1\}$, and use interpolation to define the mapping for intermediary values. This also ensures robustness to small fluctuations in the ρ_s estimate. We use $\pi(c_{\rho_s})$ to denote the resulting multinomial distribution.

Choosing A Copula Family

Given a set of M observations for two random variables X, Y , our task is now to compute the empirical multinomial signature and compare it to the template ones in order to choose the closest copula family. Let \mathcal{C} be the set of

candidate copula families from which we wish to choose the most appropriate copula. Omitting the explicit dependence on the data for readability, we use $\hat{\pi}$ and $\hat{\rho}_s$ to denote the empirical multinomial frequency over the $K \times K$ grid and the empirical Spearman’s ρ_s estimate, respectively. We choose the copula family \tilde{C} as the one that minimizes the distance between the empirical and template signatures.¹ That is:

$$\tilde{C} = \operatorname{argmin}_{C \in \mathcal{C}} d(\hat{\pi} \| \pi(c_{\hat{\rho}_s})), \quad (3)$$

where $d(\cdot \| \cdot)$ is a divergence measure between distributions. Several possible choices for this measure come to mind. The KL [Kullback and Leibler, 1951] distance is the divergence of choice between distributions, but can be sensitive to small probabilities which can occur in some of the multinomial grid cells. The L1-norm measure is less sensitive to outliers but does not measure relative deviation. In Section 5 we explore the theoretical properties of both choices, and in Section 7 we demonstrate their empirical merit.

4.2 LEARNING A TREE NETWORK

We now turn to our goal of learning the structure of a high-dimensional tree copula networks. Consider a tree structured model over N variables [Kirshner, 2007, Elidan, 2012] where the joint density can be written as

$$f_{\mathcal{X}}(x_1, \dots, x_n) = \prod_{(i,j) \in T} c_{ij}(F_i(x_i), F_j(x_j)) \prod_i f_i(x_i),$$

where c_{ij} is the copula associated with the edge (i, j) in the network. Learning the optimal tree structure can be easily carried out using a maximum spanning tree algorithm once the merit of each of the $O(N^2)$ candidate edges has been computed. However, even if we have already made the choice of the copula family for each pair of variables, we still need to estimate the maximum likelihood parameters of the copula, and then compute the maximum likelihood score. That is, taking the log of the density, for each pair of variables in the network X_i and X_j , we need to compute

$$\text{Score}(i, j) \equiv \sum_{m=1}^M \log c_{\hat{\theta}}(F_{X_i}(x_i[m]), F_{X_j}(x_j[m])), \quad (4)$$

where the sum is over instances and $\hat{\theta}$ are the maximum likelihood parameters. As we report in Section 7, the overall computations for the entire network can be demanding.

To overcome this difficulty, we again note that the precise bivariate distribution, defined via $\hat{\theta}$, is a nuisance parameter. In fact, all that we need is a proxy to the above score that will reasonably *rank* candidate edges.² Recalling that

¹In the unlikely case that more than one copula minimizes this measure, we randomly choose between the minimizing copulas.

²We note that the ρ_s -based proxy of Elidan [2012] cannot be used since it assumes the same copula for all edges.

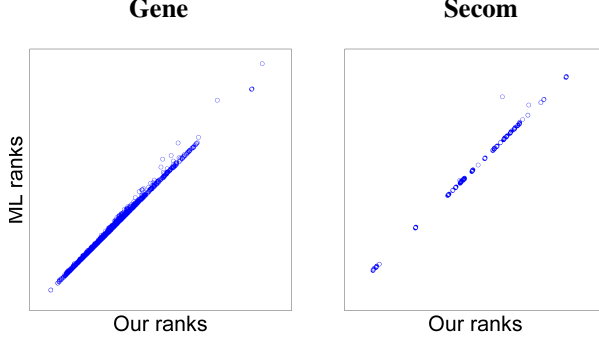


Figure 3: Edge score ranks using maximum likelihood (y-axis) vs. our multinomial proxy score (x-axis) for two of the real-life datasets used in Section 7.

our multinomial mapping roughly approximates the density, it is natural to use the implied multinomial likelihood as a proxy score. Concretely, using $C_{i,j}$ to denote the chosen copula for the variable pair X_i and X_j , instead of Eq. (4), we use

$$\text{Score}^{\text{Mult}}(i, j) = \sum_k \hat{\pi}_{i,j}[k] \log \pi(c_{i,j})[k], \quad (5)$$

where $\hat{\pi}_{i,j}$ is the empirical multinomial distribution induced by X_i and X_j .

To gauge the quality of this proxy score, Figure 3 compares the ranks of Eq. (5) and Eq. (4) for pairs of variables in two large datasets that we use in our experimental evaluation, **Gene** and **Secom** (see Section 7 for details). It is easy to see that our proxy score is near perfect for the purpose of ranking the benefit of candidate edges. Importantly, computation of $\hat{\pi}$ is linear in the (small) number of cells of the multinomial signature. Consequently, computation of our proxy score is significantly faster than the computation of the likelihood score. As we shall see in Section 7, this results in dramatic speedups of the learning procedure.

5 FINITE SAMPLE BOUNDS

Before describing how our approach can be extended to general structures, in this section we consider the theoretical properties of our building block copula selection method. Clearly, as K is increased in the multinomial mapping, we capture the density at an increasingly better granularity and, assuming continuity, asymptotic recovery follows from standard considerations. In fact, even for fixed small partitions such as $K = 4$, most standard copula families will disagree on some of the $K \times K$ multinomial bins, and asymptotic recovery can be easily guaranteed. We are more interested, however, in providing finite sample guarantees for our algorithm, both when using the Kullback-Leibler divergence and the L1-norm in Eq. (3). To the best of our knowledge ours are the first finite-sample bounds in the context of copula model selection.

We assume ρ_s is known (or has been measured) and omit it from the notation for clarity. Let \mathcal{C} be a finite copula family hypothesis class of cardinality $|\mathcal{C}| = L$ and let \mathcal{D} be a set of M i.i.d training instances sampled from $C^* \in \mathcal{C}$. Denote by $\hat{\pi}$ the empirical multinomial signature of \mathcal{D} and by $\tilde{\mathcal{C}}(\mathcal{D})$ the copula family chosen by our algorithm. The probability of mistakingly identifying the copula family is:

$$\mathbf{err}(\mathcal{D}) = P_{C^*}(\tilde{\mathcal{C}}(\mathcal{D}) \neq C^*),$$

where we use P_{C^*} as a shorthand for $P_{\mathcal{D} \sim C^*}$. We will now bound number of instances needed to ensure that the error is below a constant $\mathbf{err}(\mathcal{D}) \leq \alpha$.

5.1 KULLBACK-LEIBLER DIVERGENCE

Assume that the data was generated by a specific copula family $C_0 \in \mathcal{C}$. We will later allow C_0 to be any copula in \mathcal{C} . We start by observing that deciding between C_0 and some other $C_j \in \mathcal{C}$ based on the KL distance from $\hat{\pi}$ is equivalent to a hypothesis test:

$$\mathcal{H}_0 : \hat{\pi} \sim C_0 \quad \mathcal{H}_1 : \hat{\pi} \sim C_j,$$

where, using the likelihood ratio test, the rejection region is defined via $\lambda(C_0, C_1; \mathcal{D}) = \frac{P_{\pi(c_j)}(\mathcal{D})}{P_{\pi(c_0)}(\mathcal{D})} > 1$. Thus, classification error can be cast in terms of type I error, giving rise to a finite sample bound:

Lemma 5.1.: *Assume $\mathcal{D} \sim C_0$ or equivalently $\hat{\pi} \sim \pi(c_0)$. There exists a constant $\delta_0(j)$ such that for any $\alpha > 0$ and $C_j \in \mathcal{C}$, if $M \geq \log\left(\frac{1}{\alpha}\right) \frac{1}{\delta_0(j)}$ then*

$$P_{C_0}(d_{KL}(\hat{\pi} \parallel \pi(c_j)) \leq d_{KL}(\hat{\pi} \parallel \pi(c_0))) \leq \alpha$$

Proof: By Sanov's theorem [Cover and Thomas, 1991] we have that the type I error is $2^{-Md_{KL}(\pi_0 \parallel \pi(c_0))}$, where π_0 is the closest multinomial to $\pi(c_0)$ that is in the rejection region of the above test. π_0 is given explicitly by:

$$\pi_0[k] = \frac{\pi(c_0)[k]^\lambda \pi(c_j)[k]^{1-\lambda}}{\sum_{k'} \pi(c_0)[k']^\lambda \pi(c_j)[k']^{1-\lambda}}, \quad \lambda \in \mathbb{R},$$

where $[k]$ is the k 'th multinomial component, and λ is chosen so that $d_{KL}(\pi_0 \parallel \pi(c_0)) - d_{KL}(\pi_0 \parallel \pi(c_j)) = 0$. Taking $\delta_0(j)$ to be $d_{KL}(\hat{\pi} \parallel \pi(c_j))$ (see Cover and Thomas [1991] for details on how $\lambda, \delta_0(j)$ can be computed) we get the desired result. Note that π_0 does not depend on α . ■

Defining $\delta_0 = \min_{j \neq 0} \delta_0(j)$, we then have:

Corollary 5.2.: *Let $\mathcal{D} \sim C_0$. If $M \geq \log_2\left(\frac{L-1}{\alpha}\right) \frac{1}{\delta_0}$, then the classification error is bounded from above by α .*

Proof: Using the union bound we have:

$$\begin{aligned} P_{C_0}(\exists j : d(\hat{\pi} \parallel \pi(c_j)) \leq d(\hat{\pi} \parallel \pi(c_0))) \\ &\leq \sum_{C_j \in \mathcal{C}} P_{C_0}(d(\hat{\pi} \parallel \pi(c_j)) \leq d(\hat{\pi} \parallel \pi(c_0))) \\ &\leq \sum_{C_j \in \mathcal{C}, j \neq 0} \frac{\alpha}{L-1} = \alpha \end{aligned}$$

where, for compactness $d() \equiv d_{KL}()$. The second inequality follows from the above lemma by using $\alpha = \frac{\alpha}{L-1}$ so that $M \geq \log\left(\frac{L-1}{\alpha}\right) \frac{1}{\delta_0(j)}$ for all j . ■

Finally, we can drop the assumption that the specific generating copula family is known and, appealingly, get a bound that grows logarithmically with $\frac{1}{\alpha}$:

Theorem 5.3. : Define $\delta_C = \min_{i:C_i \in \mathcal{C}} \delta_i$. If $M \geq \log_2\left(\frac{L(L-1)}{\alpha}\right) \frac{1}{\delta_C}$ then the misclassification error is bounded from above by α .

5.2 L_1 DISTANCE

We now develop parallel bounds for the case of the L_1 -norm. Denote by $\pi(c_i)[k]$ the k -th component of the multinomial defined by C_i , and define $\delta_0(i)[k] = |\pi(c_0)[k] - \pi(c_i)[k]|$ for $C_0, C_i \in \mathcal{C}$.

Lemma 5.4.: Let $\mathcal{D} \sim C_0$, and let $\delta_0 = \min_{i \neq 0, k} \delta_0(i)[k]$. If the number of samples satisfies $M \geq \log\left(\frac{2(L-1)K^2}{\alpha}\right) \frac{1}{\delta_0^2}$, then the probability of a classification error is bounded from above by α .

Proof: For compactness define $\Delta_i[k] = |\hat{\pi}[k] - \pi(c_i)[k]|$. Then, since $\hat{\pi} \sim \pi(c_0)$, and using simple union bounds, the probability of misclassification is:

$$\begin{aligned} P_{C_0}\left(\exists i \neq 0 : \sum_k \Delta_i[k] \leq \sum_k \Delta_0[k]\right) \\ \leq \sum_{i \neq 0} P_{C_0}\left(\sum_k \Delta_i[k] \leq \sum_k \Delta_0[k]\right) \\ \leq \sum_{i \neq 0, k} P_{C_0}(\Delta_i[k] \leq \Delta_0[k]) \end{aligned}$$

Next, by definition $\Delta_i[k] \leq \Delta_0[k] \Leftrightarrow \Delta_0[k] \geq \delta_0(i)[k]$. Also, since $\mathcal{D} \sim C_0$, we have $E(\hat{\pi}[k]) = \pi(c_0)[k]$. Using Hoeffding's inequality we then have:

$$\begin{aligned} P_{C_0}(\Delta_i[k] \leq \Delta_0[k]) &= P_{C_0}(\Delta_0[k] \geq \delta_0(i)[k]) \\ &\leq 2e^{-2M\delta_0^2}. \end{aligned}$$

If we now choose the number of samples to be $M \geq \log\left(\frac{2(L-1)K^2}{\alpha}\right) \frac{1}{\delta_0^2}$, the result easily follows. ■

Now, using a similar argument to the KL case, we have

Theorem 5.5. : Define $\delta_C = \min_i \delta_i$. For all α , if $M \geq \log_2\left(\frac{L(L-1)K^2}{\alpha}\right) \frac{1}{\delta_C^2}$, then the misclassification error is bounded from above by α .

As an example consider sample data that is distributed according to AMH copula with Spearman's rho equals 0.6. Assuming the copula hypothesis class consists of AMH, Clayton, Gumbel and Plackett copulas. Then using $k = 2$, it is easily verified that $\delta_0 = 0.0713$. Thus, according to 5.4, in order to bound the classification error by $\alpha = 0.05$, at least 1217 samples are needed.

6 LEARNING GENERAL NETWORKS

We now show how our structure learning approach of Section 4 can be adapted to the more elaborate task of learning a copula graphical model with a general structure. As is commonly done, due to the super-exponential nature of the search space, we learn the structure via a greedy search procedure that involves local modifications to the structure (e.g., add/delete/reverse an edge). Similarly to the case of trees, we start by generalizing the local copula selection building block and then explain how this can be used when learning a global structure.

6.1 CHOOSING THE COPULA

Recall that in order to choose a copula family in the bivariate case, we first evaluate the empirical measure of association $\hat{\rho}_s$, as well as the bivariate statistics of the data, and then choose the copula family signature that is closest to the empirical distribution for $\hat{\rho}_s$. In a nutshell, we will choose a copula family for more than two variables by aggregating bivariate distances.

Before doing so, however, we need to evaluate $\hat{\rho}$. For the Gaussian copula, our path is obvious since each bivariate marginal is characterized by its own dependence parameters $\Sigma_{i,j}$, and the corresponding measure of association $\hat{\rho}_{i,j}$ can be computed as before. However, the situation is quite different for other copula families. For example, all bivariate marginals of an n -dimensional Archimedean copula have the *same* dependence parameter so that we require $\hat{\rho}_{i,j} = \hat{\rho}$ for all i, j . Thus, a natural choice in the common case of a *one parameter* family is to estimate $\hat{\rho}$ using

$$\hat{\rho} = \sum_{X_i, X_j, i < j} \binom{n}{2}^{-1} \hat{\rho}_{X_i, X_j}.$$

Note that this is one of the standard generalizations of Spearman's rho [Schmid and Schmidt, 2007]. Then, with $\hat{\rho}_{i,j} = \hat{\rho}$ in hand, we select the copula family that minimizes the sum of distances between the empirical and template multinomials signatures, similarly to Eq. (3):

$$\tilde{C} = \operatorname{argmin}_{C_{\hat{\rho}}} \sum_{X_i, X_j, i < j} d(\hat{\pi} \| \pi(c_{\hat{\rho}_{i,j}})).$$

6.2 EVALUATING THE STRUCTURE SCORE

As in bivariate case, after choosing the copula family, we still face the challenge of comparing the benefit of different candidate structural changes. Concretely, using Eq. (2), we we need to evaluate the conditional likelihood score:

$$\begin{aligned} \text{Score}(i, \mathbf{P}_{\mathbf{a}_i}) &\equiv \\ &\sum_{m=1}^M \log \frac{c_{\hat{\theta}}(F_i(x_i[m]), \{F_j(x_j[m])\}_{j \in \mathbf{P}_{\mathbf{a}_i}})}{c_{\theta}(\{F_j(x_j[m])\}_{j \in \mathbf{P}_{\mathbf{a}_i}})}, \end{aligned}$$

	N	F	G	C	A	M
N	0.87	0.06	0.02	0.00	0.02	0.02
F	0.05	0.89	0.01	0.00	0.02	0.04
G	0.01	0.01	0.98	0.00	0.00	0.00
C	0.00	0.00	0.00	0.97	0.02	0.01
A	0.02	0.02	0.00	0.03	0.92	0.01
M	0.02	0.03	0.00	0.00	0.02	0.94

HELM

	N	F	G	C	A	M
N	0.93	0.01	0.01	0.00	0.03	0.03
F	0.14	0.74	0.00	0.00	0.04	0.08
G	0.13	0.00	0.87	0.00	0.00	0.00
C	0.13	0.00	0.00	0.84	0.03	0.00
A	0.01	0.01	0.00	0.00	0.96	0.02
M	0.00	0.00	0.00	0.00	0.02	0.98

Huard

	N	F	G	C	A	M
N	0.96	0.02	0.00	0.00	0.00	0.01
F	0.02	0.94	0.00	0.00	0.00	0.048
G	0.00	0.00	1.00	0.00	0.00	0.00
C	0.00	0.00	0.00	1.00	0.00	0.00
A	0.04	0.12	0.00	0.02	0.79	0.02
M	0.02	0.03	0.00	0.00	0.00	0.98

Costly ML

Figure 4: Copula family selection performance for synthetically generated data with 1000 samples. Methods compared are our HELM, that of Huard [Huard et al., 2006], and time consuming ML estimation. Each confusion matrix shows the percentage of the predicted family (columns) given the generating family (rows).

where $\hat{\theta}$ are the maximum likelihood parameters of the copula associated with X_i and its parents.

Once again, we face the bottleneck of maximum likelihood estimation. Whenever an analytically simple relationship between $\hat{\rho}$ (or Kendall’s τ) and $\hat{\theta}$ exists, we use the heuristic proposed by [P. Embrechts and M. Hofert, 2010] and simply invert the average association measure described above. For other copula families (e.g. Ali-Mikhail), we resort to a standard optimization procedure such as conjugate gradient. Note that even in this case, we perform costly estimation *only* for the chosen copula family, and are thus still significantly more efficient than a full maximum likelihood selection and estimation procedure.

6.3 ADJUSTING THE SCALE OF THE SCORE

To learn a structure that allows for several parents for each variable, all family scores must obviously lie on the same scale. However, the proxy score we use in the case of a single parent is based on a discrete multinomial likelihood (Eq. (5)), while for multiple parents we use a real-valued conditional likelihood (Eq. (6)). Thus, to rank candidate structural changes, we must somehow calibrate these scores relative to each other.

Fortunately, as is clearly evident in Figure 3, our single parent *proxy* scores are almost linearly correlated to the *exact* maximum likelihood scores. Consequently, all that is required in order to accurately approximate the needed scores is to recover this linear transformation via straightforward regression. Concretely, we randomly choose few (e.g. 10%) of the variables pairs, and solve the following regression problem:

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} \sum_{i,j} (\operatorname{Score}(i,j) - \beta_0 - \beta_1 \operatorname{Score}^{\text{Mult}}(i,j))^2$$

We then use $\hat{\beta}$ to calibrate our bivariate scores:

$$\widetilde{\operatorname{Score}}(i,j) = \beta_0 + \beta_1 \cdot \operatorname{Score}^{\text{Mult}}(i,j).$$

To summarize: starting with the empty graph G_\emptyset , we rank

the different $O(N^2)$ candidate edges using our multinomial approximation score. Next, we calibrate these scores using the regression coefficients β . These calibrated score are then used together with the multi-parent scores to guide the greedy structure learning procedure.

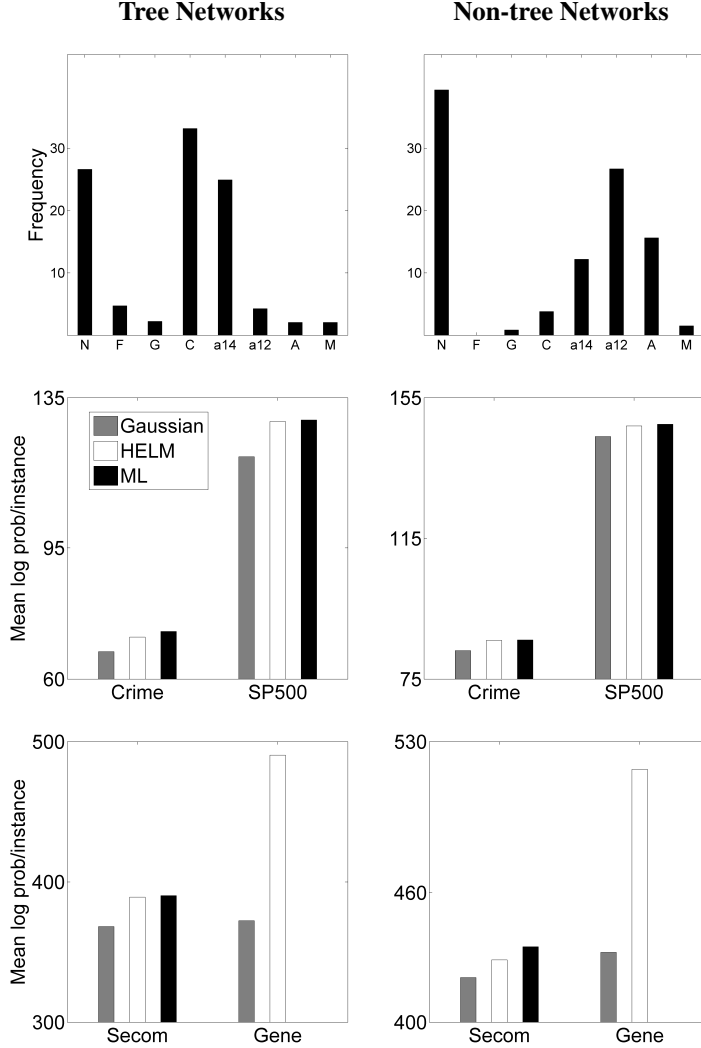
7 EXPERIMENTAL EVALUATION

We now evaluate the ability of our **HELM** approach to efficiently learn expressive copula networks that generalize well. We start by evaluating the merit of the **HELM** model selection building block in the case where the generating distribution is known. We then demonstrate the power of **HELM** when learning high-dimensional structures for sizeable real-life domains.

7.1 COPULA MODEL SELECTION

To evaluate our **HELM** copula model selection building block, we synthetically generate i.i.d. instances from different copula families and attempt to identify the generating family from the samples. We compare our **HELM** approach to the standard maximum likelihood (**ML**) approach (using an inversion of the empirical Kendall tau or Spearman’s rho for fast estimation where possible), and to a Bayesian approach from the copula community suggested by **Huard** [Huard et al., 2006].

Similarly to Huard et al. [2006], we consider a collection of copula families that exhibit varied dependence patterns: Normal (**N**), Frank (**F**), Gumbel (**G**), Clayton (**C**), Ali-Mikhail-Haq (**A**), and Farlie-Gumbel-Morgenstern (**M**) (see [Joe, 1997, Nelsen, 2007] for details of these copulas). For each family, we precompute its multinomial signature as described in Section 4. To cover a wide range of dependence levels, we generate the synthetic data as follows: for values of Spearman’s ρ_s ranging from 0.25 to 0.95, we randomly choose a copula family C , and generate $M = 1000$ i.i.d samples from C_{ρ_s} . We repeat this 1000 times for each value of ρ_s and use the different methods to predict the generating copula family.



Tree Networks

	Gaussian copula	Our method	ML	Speed Factor
Crime	67.31 (0.84)	71.2 (0.79)	72.75 (0.76)	175
Secom	368.19 (3.68)	389.12 (3.73)	390.36 (3.58)	182
SP500	119.36 (3.15)	128.73 (3.18)	129.15 (3.26)	307
Gene	372.43 (4.61)	490.56 (4.83)	–	∞

Non-tree Networks

	Gaussian copula	Our Method	ML	Speed Factor
Crime	83.16 (0.92)	85.79 (0.86)	86.12 (0.89)	78
Secom	420.72 (3.35)	428.96 (3.83)	435.74 (3.79)	76.3
SP500	144.27 (3.45)	146.94 (3.38)	147.52 (3.41)	59.9
Gene	432.42 (4.72)	517.25 (4.63)	–	∞

Figure 5: Comparison of copula networks learned using the different methods for tree and non-tree networks. **(left top)** shows the distribution of the chosen copula families when learning using standard maximum likelihood (**ML**) for the **Secom** dataset. **(left middle/bottom)** summarizes the average test log-probability per instance performance of our method (white bars), **ML** (black bars) and the Gaussian copula baseline (gray bar). **(right)** detailed generalization performance along with standard deviation across random folds (in parentheses) and speedup factor of our method relative to **ML**.

In Figure 4 we report average results in the form of confusion matrices that show the distribution of the predicted copula family (columns) for each generating copula family (rows). Results for our approach are with $K = 8$ and using KL (results were qualitatively similar using $K = 4$ and the L_1 distance). As can be seen, **HELM** surpasses **Huad** on average and is not far beyond the much slower **ML**. This is to be expected since we intentionally took a crude but efficient view of the distribution, a crucial step toward the goal of performing global structure learning.

The above competitiveness comes with substantial computational advantages. A single model selection task when using **ML** took 1.08×10^{-2} seconds on average. Using **HELM** this took only 1.1545×10^{-4} seconds on average,

a two orders of magnitude speedup. Advantages are even greater for $K = 4$ since **HELM** is quadratic in K . In this case, **HELM** is close to 200 faster than **ML**, while suffering negligible decrease in predictive performance. Finally, while **Huad** does reasonably well in terms of predictive performance, this comes at an enormous computational cost, taking an average of 0.28 seconds, or 4 orders of magnitude slower than **HELM**.

7.2 LEARNING EXPRESSIVE NETWORKS

We now evaluate the merit of **HELM** for learning expressive copula networks for real-life domains that benefit from a rich mix of local representations. We consider four real-

life datasets that are quite sizable in the context of structure learning of non-Gaussian real-valued models:

- **Crime** (UCI repository). 1994 instances of **100** census variables ranging from household size to fraction of children born out of marriage, for 1994 U.S. communities.
- **Secom** (UCI repository). 1567 instances of **362** variables collected from sensors during a semi-conductor manufacturing process, corresponding to key factors that effect downstream yield.
- **SP500**. End of day changes of the **500** Standard and Poor's index stocks (variables) over a period of close to 2000 trading days (samples).
- **Gene**. A compendium of gene expression. We focus on **999** genes (variables) that have at most one missing experiment, resulting in 2000 samples.

For each domain, we learn a copula network model using **HELM** as well as using standard maximum likelihood (using fast inversion of ρ_s or τ_K where possible). In both cases, we allow for a mix of Gaussian, Frank, Gumbel, Clayton, arch12, arch14, Ali-Mikhail and FGM copulas. To make comparison to the costly ML feasible, we learn networks with up to two parents. For the univariate marginals for both methods, we use a standard kernel-based approach [Parzen, 1962] with the common Gaussian kernel (see, for example, [Bowman and Azzalini, 1997] for details). As an additional baseline, we also consider learning only with a Gaussian copula, which is the strongest of all single family baselines. Finally, we note that due to its significant computational demands, the Bayesian method of **Huad** could not be used in these experiments.

We start by qualitatively demonstrating the real-life need for expressive modeling, or for the combination of different local representations within the same model. As an example Figure 5(left top) shows the distribution of the copula families chosen when learning a mixed model using the **ML** method for the **Secom** dataset. Obviously, the learned model is a rich one.

Quantitatively, Figure 5(left middle/bottom) shows that a mixed **ML** model (black bar) also leads to better generalization relative to the best single family baseline (gray bar) in terms of test set log-probability per instance. Also shown is the performance of **HELM** (white bar). As can be seen, **HELM** is competitive with the costly **ML** method. The table on the right includes the average test performance results along with standard deviations (in parentheses) across 10 folds. Importantly, note that the improvement over the single family baseline is significant since the scale of improvement is in bits per instances. Thus, an improvement of, for example, 10 bits per instance is equivalent to each test instance being on average 2^{10} more likely.

Recall that our goal was not simply to learn competitive expressive networks but to do so highly efficiently so as to facilitate scaling up of structure learning. Speed up factors of **HELM** relative to **ML** are reported in the right-hand column of the tables in Figure 5. As can be seen, the runtime improvements are dramatic at over two orders of magnitudes when learning tree networks. To make these numbers concrete, for example, using **HELM** to learn a mixed tree for the **SP500** domain took less than a minute, while for **ML** the average runtime was nearly 5 hours. For the **Gene** data set with 1000 variables, although learning a mixed network with **HELM** took only around 4.5 minutes, **ML** did not terminate after two days. A substantial runtime improvement is also evident for more general structures. For example, learning using **HELM** took around an hour and a half for **SP500**, while learning using **ML** took over three days. Dramatically, although **HELM** was able to learn a mixed **Gene** network model in less than two hours, learning a model for this domain using **ML** proved impractical, and did not terminate within a week.

8 SUMMARY

We presented **HELM**, an algorithm for efficiently learning copula networks that allows for a rich mix of varied copula families within the *same* model. We demonstrated the substantial computational advantages of using our multinomial signature based approach when learning complex models for several varied sizeable real-life domains.

Our contribution is three fold. First, we presented a straightforward but powerful copula model selection building block that, even in the simple bivariate case, is competitive with maximum likelihood and other estimation approaches while offering dramatic runtime improvements. We further derive finite-sample guarantees for this building block. To the best of our knowledge, these are the first such guarantees in the context of copula model selection.

Second, we showed how our building block gives rise accurate and efficient *ranking* of candidate structures, resulting in highly efficient global structure learning. Third, the computational advantages allows us to scale up structure learning and easily cope with domains that are prohibitive if tackled using standard procedures. Indeed, to the best of our knowledge, ours is the first structure learning method that allows for a mix of local real-valued representations and that has been applied to domains of this size.

Acknowledgements

This work was supported in part by an ISF Center of Excellence grant, by an Israeli Ministry of Science center of knowledge and by the Intel Collaborative Research Institute for Computational Intelligence (ICRI-CI). We thank Elad Eban and Elad Mezuman for their valuable comments on different drafts of this work.

References

- A. Bowman and A. Azzalini. *Applied Smoothing Techniques for Data Analysis*. Oxford University Press, 1997.
- T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, New York, 1991.
- G. Elidan. Lightning-speed structure learning of nonlinear continuous networks. In *Proceedings of the AI and Statistics Conference (AISTATS)*, 2012.
- Gal Elidan. Copula Bayesian networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.
- J.-D. Fermanian. Godness-of-fit tests for copulas. *Journal of Multivariate Analysis*, 95:52–119, 2005.
- Christian Genest and L.-P. Rivest. Statistical inference procedures for bivariate archimedean copulas. *Statist.Assoc.*, 88(8):1034 – 1043, 1993.
- Christian Hering and Marius Hofert. Godness-of-fit tests for archimedean copulas in large dimensions. *Working paper*, 2010.
- David Huard, Guillaume Ivin, and Anne-Catherine Favre. Bayesian copula selection. *Comput. Stat. Data Anal.*, 51(2):809–822, 2006.
- H. Joe. Multivariate models and dependence concepts. *Monographs on Statistics and Applied Probability*, 73, 1997.
- A. Justel, D. Pena, and R. Zamar. A multivariate kolmogorov-smornov test of goodness of fit. *Statistical Probability Letters*, 35:251–259, 1997.
- S. Kirshner. Learning with tree-averaged densities and distributions. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.
- S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:76–86, 1951.
- R. Nelsen. *An Introduction to Copulas*. Springer, 2007.
- E. Parzen. On estimation of a probability density function and mode. *Annals of Math. Statistics*, 33:1065–1076, 1962.
- J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.
- P. Embrechts and M. Hofert. Statistical inference for copulas in high dimension: a simulation study. *Statistical Probability Letters*, 35:251–259, 2010.
- Friedrich Schmid and Rafael Schmidt. Multivariate extensions of spearman’s rho and related statistics. *Statistics and Probability Letters*, 77(4):407 – 416, 2007.
- A. Sklar. Fonctions de repartition a n dimensions et leurs marges. *Publications de l’Institut de Statistique de L’Universite de Paris*, 8:229–231, 1959.
- Y. Tenzer and G. Elidan. Speedy model selection (SMS) for copula models. In *Uncertainty in Artificial Intelligence (UAI)*, 2013.
- Andrew Wilson and Zoubin Ghahramani. Copula processes. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.

Metrics for Probabilistic Geometries

Alessandra Tosi
Dept. of Computer Science
Universitat Politècnica
de Catalunya
Barcelona, Spain

Søren Hauberg
DTU Compute
Technical University
of Denmark
Denmark

Alfredo Vellido
Dept. of Computer Science
Universitat Politècnica
de Catalunya
Barcelona, Spain

Neil D. Lawrence*
Dept. of Computer Science
The University
of Sheffield
Sheffield, UK

Abstract

We investigate the geometrical structure of probabilistic generative dimensionality reduction models using the tools of Riemannian geometry. We explicitly define a distribution over the natural metric given by the models. We provide the necessary algorithms to compute expected metric tensors where the distribution over mappings is given by a Gaussian process. We treat the corresponding latent variable model as a Riemannian manifold and we use the expectation of the metric under the Gaussian process prior to define interpolating paths and measure distance between latent points. We show how distances that respect the expected metric lead to more appropriate generation of new data.

1 MOTIVATION

One way of representing a high dimensional data set is to relate it to a lower dimensional set of *latent variables* through a set of (potentially nonlinear) functions. If the i th data point and the j th feature is represented by $y_{i,j}$, it might be related to a q dimensional vector of latent variables $\mathbf{x}_{i,:}$ as

$$y_{i,j} = f_j(\mathbf{x}_{i,:}) + \epsilon_i,$$

where $f_j(\cdot)$ is a nonlinear function mapping to the j th feature of the data set and ϵ_i is a noise corruption of the underlying function. A manifold derived from a finite data set can never be precisely determined across the entire input range of \mathbf{x} . We consider posterior distributions defined over $f_j(\cdot)$ and we focus on the uncertainty defined over the local metric of the manifold itself. This allows us to define distances that are based on metrics that take account of the uncertainty with which the manifold is defined. We use these metrics to define distances between points in the latent space that respect these metrics.

*Also at Sheffield Institute for Translational Neuroscience, SITraN. Sheffield, UK

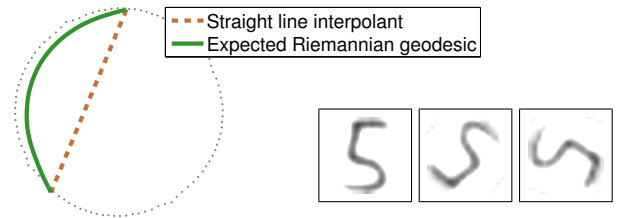


Figure 1: The latent space from a GP-LVM that was trained over a dataset of artificially rotated digits. Black dots represent the latent points. The dashed brown line show the commonly used straight-line interpolant, and the green curve is the suggested expected Riemannian geodesic. This figure is best viewed in colour.

When the mappings $f_j(\cdot)$ are nonlinear, the latent variable model (LVM) can potentially capture non-linearities on the data and thereby provide an even lower dimensional representation as well as a more useful view of the data. While this line of thinking is popular, it is not without its practical issues. As an illustrative example, Fig. 1 shows the latent representation of a set of artificially rotated images obtained through a Gaussian process latent variable model (GP-LVM). It is clear from the display that the latent representation captures the underlying periodic structure of the process which generated the data (a rotation). If we want to analyse the data in the latent space, e.g. by interpolating latent points, our current tools are insufficient. As can be seen, fitting a straight line in the latent space between the two-points leads to a solution that does not interpolate well in the data space: the interpolant goes through regions where the data does not reside, regions where the actual functions, $f_j(\cdot)$, cannot be well determined.

This observation raises several related questions about the choice of interpolant: 1) what is the natural choice of interpolant in the latent space? And, 2) if the natural interpolant is not a straight line, are Euclidean distances still meaningful? We answer these questions for the GP-LVM, though our approach is applicable to other generative models as

well. We consider here a metric which reflects the intrinsic properties of the original data and recovers some information loss due to the nonlinear mapping performed by the model. We find that for smooth LVMs the metric from the observation space can be brought back to the latent space in the form of a random Riemannian metric. We then provide algorithms for computing distances and shortest paths (geodesics) under the expected Riemannian metric. With this the natural interpolant becomes a curve, which follows the trend of the data.

Overview In Section 2 we introduce the concepts of Riemannian geometry, the tool on which we rely on later on in the paper. Section 3 provides an overview of the state of the art in probabilistic dimensionality reduction, introducing the class of models to which the proposed methodology can be extended. In Section 4 we use the probabilistic nature of the generative LVMs to explicitly provide distributions over the metric tensor; first, we provide the general expressions, then we specialise these to the GP-LVM as an example. Finally, we show how to compute shortest paths (geodesics) over the latent space. Experimental results are provided in Section 5, and the paper is concluded with a discussion in Section 6.

2 CONCEPTS OF RIEMANNIAN GEOMETRY

We study latent variable models (LVMs) as embeddings of uncertain surfaces (or manifolds) into the observation space. From a machine learning point of view, we can interpret this embedded manifold as the underlying support of the data distribution. To this end, we review the basic ideas of differential geometry, which study surfaces through local linear models.

Gauss’ study [1827] of curved surfaces are among the first examples of (deterministic) LVMs. He noted that a q -dimensional surface embedded in a p -dimensional Euclidean space¹ is well-described through a mapping $f : \mathbb{R}^q \rightarrow \mathbb{R}^p$. The q -dimensional representation of the surface is known as the *chart* (in machine learning terminology, this corresponds to the *latent space*). In general, the mapping f between the chart and the embedding space is not *isometric*, e.g. the Euclidean length of a straight line l in the chart does not match the length of the embedded curve $f(l)$ as measured in the embedding space. Intuitively, the chart provides a distorted view of the surface (see Fig. 2 for an illustration). To rectify this view, Gauss noted that the

¹Historically, Gauss considered the case of two-dimensional surfaces embedded in \mathbb{R}^3 , while the extension to higher dimensional *manifolds* is due to Bernhard Riemann.

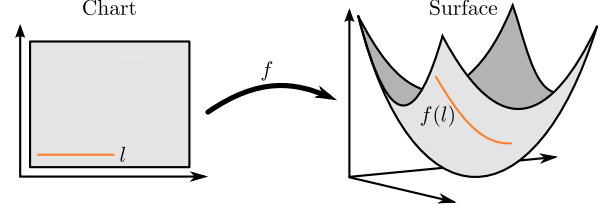


Figure 2: An illustration of the standard surface model; f maps the chart into the embedding space.

length of a curve is

$$\text{Length}(f(l)) = \int_0^1 \left\| \frac{\partial f(l(t))}{\partial t} \right\| dt = \int_0^1 \left\| \mathbf{J} \frac{\partial l(t)}{\partial t} \right\| dt, \quad (1)$$

where \mathbf{J} denotes the Jacobian of f , i.e.

$$[\mathbf{J}]_{i,j} = \frac{\partial f_i}{\partial l_j}. \quad (2)$$

Measurements on the surface can, thus, be computed in the chart locally, and integrated to provide global measures. This gives rise to the definition of a *local* inner product, known as a *Riemannian metric*.

Definition (Riemannian Metric). A Riemannian metric \mathbf{G} on a manifold \mathcal{M} is symmetric and positive definite matrix which defines a smoothly varying inner product

$$\langle \mathbf{a}, \mathbf{b} \rangle_x = \mathbf{a}^\top \mathbf{J}^\top \mathbf{J} \mathbf{b} = \mathbf{a}^\top \mathbf{G}(x) \mathbf{b} \quad (3)$$

in the tangent space $T_x \mathcal{M}$, for each point $x \in \mathcal{M}$ and $\mathbf{a}, \mathbf{b} \in T_x \mathcal{M}$. The matrix \mathbf{G} is called the *metric tensor*.

Remark The Riemannian metric need not be restricted to $\mathbf{G} = \mathbf{J}^\top \mathbf{J}$ and can be any smoothly changing symmetric positive definite matrix [do Carmo, 1992]. We restrict ourselves to the more simple definition as it suffices for our purposes, but note that the more general approach has been used in machine learning, e.g. in *metric learning* [Haugberg et al., 2012] and *information geometry* [Amari and Nagaoka, 2000].

From this definition, Eq. 1 reduces to

$$\text{Length}(\gamma) = \int_0^1 \sqrt{\langle \gamma'(t), \gamma'(t) \rangle_{\gamma(t)}} dt \quad (4)$$

for a general curve $\gamma : [0, 1] \rightarrow \mathbb{R}^p$.

Definition (Geodesic curve). Given two points $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{M}$, a *geodesic* is a length-minimising curve connecting the points

$$\gamma_g = \arg \min_{\gamma} \text{Length}(\gamma), \quad \gamma(0) = \mathbf{x}_1, \gamma(1) = \mathbf{x}_2. \quad (5)$$

It can be shown [do Carmo, 1992] that geodesics satisfy the following second order ordinary differential equation

$$\gamma'' = -\frac{1}{2}\mathbf{G}^{-1} \left[\frac{\partial \text{vec } \mathbf{G}}{\partial \gamma} \right]^\top (\gamma' \otimes \gamma'), \quad (6)$$

where $\text{vec } \mathbf{G}$ stacks the columns of \mathbf{G} and \otimes denotes the Kronecker product. The Picard-Lindelöf theorem [Tenenbaum and Pollard, 1963] then implies that geodesics exist and are locally unique given a starting point and an initial velocity.

3 PROBABILISTIC DIMENSIONALITY REDUCTION

Nonlinear dimensionality reduction methods [Lee and Verleysen, 2007] provide a flexible data representation which can provide a more faithful model of the observed multivariate datasets than the linear ones. One approach is to perform probabilistic nonlinear dimensionality reduction defining a model that introduces a set of unobserved (or latent) variables \mathbf{X} that can be related to the observed ones \mathbf{Y} in order to define a joint distribution over both. These models are known as latent variable models (LVMs). The latent space is dominated by a prior distribution $p(\mathbf{X})$ which induces a distribution over \mathbf{Y} under the assumption of a probabilistic mapping

$$y_{i,j} = f_j(\mathbf{x}_i) + \epsilon_i, \quad (7)$$

where \mathbf{x}_i is the latent point associated with the i^{th} observation \mathbf{y}_i , j is the index of the features of \mathbf{Y} , and ϵ_i is a noise term, accounts for both noise in the data as well as for inaccuracies in the model. The noise is typically chosen as Gaussian distributed $\epsilon \sim \mathcal{N}(0, \beta^{-1})$, where β is the precision.

One of the advantages of this approach is that it accommodates dimensionality reduction in an intuitive manner, if we assume that the dimensionality of the latent space is significantly lower than that of the observation space. In this case, the reduced dimensionality provides us with both implicit regularisation and a low-dimensional representation of the data, which can be used for visualisation (and, therefore, for data exploration [Vellido et al., 2011]) if the dimension is low enough.

If the mapping $f = W$ is taken to be linear:

$$y_{i,j} = \mathbf{w}_j^\top \mathbf{x}_i + \epsilon_i, \quad (8)$$

and the prior $p(\mathbf{X})$ to be Gaussian, this model is known as probabilistic principal component analysis [Tipping and Bishop, 1999]. The conditional probability of the data given the latent space can be written as

$$p(\mathbf{y}_i | \mathbf{x}_i, \mathbf{W}, \beta) = \mathcal{N}(\mathbf{y}_i | \mathbf{W}\mathbf{x}_i, \beta^{-1}\mathbf{I}). \quad (9)$$

With a further assumption of independence across data points, the marginal likelihood of the data is

$$p(\mathbf{Y}) = \int \prod_{i=1}^N p(\mathbf{y}_i | \mathbf{x}_i, \mathbf{W}, \beta) p(\mathbf{x}_i) d\mathbf{X}. \quad (10)$$

In general, this approach can be applied to both linear and nonlinear dimensionality reduction models, leading to the definition of, for instance, Factor Analysis [Bartholomew, 1987], Generative Topographic Mapping (GTM) [Bishop et al., 1998], or Gaussian Process-LVM (GP-LVM) [Lawrence, 2005] to name a few.

One example that generalises from the linear case to the nonlinear one is the GTM, in which the noise model is taken to be a linear combinations of a set of M basis functions

$$y_{i,j} = \sum_{m=1}^M \mathbf{w}_j^\top \phi_m(\mathbf{x}_i) + \epsilon_i. \quad (11)$$

This model can be seen as a mixture of distributions (usually Gaussian radial basis distributions) whose centres are constrained to lay on an intrinsically low-dimensional space. These centres can be interpreted as data prototypes or cluster centroids that can be further agglomerated in a full blown clustering procedure. In this manner, GTM mixes the functionalities of Self-Organising Maps and mixture models by providing both data visualisation over the latent space and data clustering [Olier and Vellido, 2008]. If the prior over the latent space is chosen to be Gaussian, this model leads, in a similar way of probabilistic PCA, to a Gaussian conditional distribution of the data

$$p(\mathbf{y}_i | \mathbf{x}_i, \mathbf{W}, \beta) = \mathcal{N} \left(\mathbf{y}_i \left| \sum_{m=1}^M \mathbf{w}_j^\top \phi_m(\mathbf{x}_i), \beta^{-1}\mathbf{I} \right. \right). \quad (12)$$

In the classic approach the latent variables are marginalised and the parameters are optimised by maximising the model likelihood. An alternative (and equivalent) approach proposes to marginalise the parameters and optimise the latent variables, leading to Gaussian Process Latent Variables Model (GP-LVM).

In terms of applications, Grochow et al. [2004] animate human poses using *style-based inverse kinematics* based on a GP-LVM model. The animation is performed under a prior towards small Euclidean motions in the latent space, i.e. under the same assumptions as those leading to a straight-line interpolant. As the Euclidean metric does not match that of the observation space, this prior is difficult to interpret. In a related application, Urtasun et al. [2005] track the pose of a person in a video sequence with a similar prior and, hence, similar considerations hold. Recently, Gonczarek and Tomczak [2014] track human poses in images under a Brownian motion prior in the latent space. Again, this relies on a meaningful metric in the latent space.

In all of the above application, it is beneficial if the metric in the latent space is related to that of the observation metric.

4 METRICS FOR PROBABILISTIC LVMs

The common approach to estimate local metrics rely on assumptions over the neighbourhoods defined in the observed space (see e.g. [Hastie and Tibshirani, 1996, Ramanan and Baker, 2011]). This might be less efficient in presence of high dimensional noise, because the induced distances may not be reliable. One way to deal with this problem is to define a noise model (7) and to assume a global belief over the geometry of the data. This way, the resulting models have the advantage of providing an intrinsic local metric which is able to deal with noise.

In this paper we only consider smooth generative models for manifold learning. This contrasts with prior approaches such as [Bregler and Omohundro, 1994, Tenenbaum, 1997, Tenenbaum et al., 2000] that use metrics which vary discretely across the space (see also [Lawrence, 2012] for relations to Gaussian models).

We define here the local metric tensor for generative LVMs. We then illustrate the specific case of GP-LVM, providing an algorithm to compute shortest path.

4.1 THE DISTRIBUTION OF THE NATURAL METRIC

When the mapping f in Eq. 7 is differentiable, it can be interpreted as the mapping between the *chart* (or *latent space*) and the embedding space (c.f. Section 2). Then it is possible to explicitly compute the natural Riemannian metric of the given model.

Let \mathbf{J} be the Jacobian (as in Eq. 2), then the tensor

$$\mathbf{G} = \mathbf{J}^\top \mathbf{J}$$

defines a local inner product structure over the latent space according to Eq. 3.

In the case of LVMs where the conditional probability over the Jacobian follow a Gaussian distribution, this naturally induces a distribution over the local metric tensor \mathbf{G} . Assuming independent rows of \mathbf{J}

$$p(\mathbf{J} | \mathbf{X}, \mathbf{f}, \beta) = \prod_{j=1}^p \mathcal{N}(\mu_{J(j,:)}, \Sigma_J), \quad (13)$$

the resulting random variable follow a non-central Wishart distribution [Anderson, 1946]:

$$\mathbf{G} = \mathcal{W}_q(p, \Sigma_J, \mathbb{E}[\mathbf{J}^\top] \mathbb{E}[\mathbf{J}]), \quad (14)$$

where p represents the number of degrees of freedom; the quantity $\Sigma_J^{-1} \mathbb{E}[\mathbf{J}^\top] \mathbb{E}[\mathbf{J}]$ is known as the non-centrality ma-

trix and it is equal to zero in the central Wishart distribution. The Wishart distribution is a multivariate generalisation of the Gamma distribution.

4.2 GP-LVM LOCAL METRIC

A Gaussian Process (GP) is used to describe distributions over functions and it is defined as a collection of random variables, any finite number of which have a joint Gaussian distribution [Rasmussen and Williams, 2006]. Given a vector $\mathbf{x} \in \mathbb{R}^q$, a GP determined by its mean function and its covariance function is denoted $f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$. From this, it is possible to generate a random vector \mathbf{f} which is Gaussian distributed with covariance matrix given by $(\mathbf{K})_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$.

Gaussian Processes have been used in probabilistic nonlinear dimensionality reduction to define a prior distribution over the mapping f (in Eq. 7), leading to the formulation of GP-LVM. This way, the likelihood of the data \mathbf{Y} given \mathbf{X} is computed by marginalising the mapping and optimising the latent variables:

$$p(\mathbf{Y} | \mathbf{X}, \mathbf{f}, \beta) = \prod_{j=1}^p \mathcal{N}(\mathbf{y}_{:,j}, \mathbf{K} + \beta^{-1} \mathbf{I}) = \prod_{j=1}^p \mathcal{N}(\mathbf{y}_{:,j}, \tilde{\mathbf{K}}). \quad (15)$$

To follow the notation introduced in Section 3, the noise model is defined by

$$y_{i,j} = \mathbf{K}_{(\mathbf{x}_i, \mathbf{x})} \mathbf{K} \mathbf{Y}_{:,j} + \epsilon_i, \quad (16)$$

Due to the linear nature of the differential operator, the derivative of a Gaussian process is again a Gaussian process ([Rasmussen and Williams, 2006] §9.4), as long as the covariance function is differentiable. This property allows inference and predictions about derivatives of a Gaussian Process, therefore the Jacobian \mathbf{J} of the GP-LVM mapping can be computed over continuum for every latent point \mathbf{x}_* and we denote with $\frac{\partial \mathbf{y}_*}{\partial x^{(i)}}$ the partial derivative of $\mathbf{y}(x_*)$ with respect to the i^{th} component in the latent space. We call $\mathbf{J}^\top = \frac{\partial \mathbf{y}_*}{\partial \mathbf{x}} = \left[\frac{\partial \mathbf{y}_*}{\partial x^{(1)}}; \dots; \frac{\partial \mathbf{y}_*}{\partial x^{(q)}} \right]$, where $\frac{\partial \mathbf{y}_*}{\partial \mathbf{x}}$ is a $q \times p$ matrix whose columns are multivariate normal distributions. We now consider the jointly Gaussian random variables

$$\begin{bmatrix} \mathbf{Y} \\ \frac{\partial \mathbf{y}_*}{\partial \mathbf{x}} \end{bmatrix} \sim \mathcal{N} \left(\mathbf{0}, \begin{bmatrix} \tilde{\mathbf{K}}_{\mathbf{x}, \mathbf{x}} & \partial \tilde{\mathbf{K}}_{\mathbf{x}, *} \\ \partial \tilde{\mathbf{K}}_{\mathbf{x}, *}^\top & \partial^2 \tilde{\mathbf{K}}_{*,*} \end{bmatrix} \right), \quad (17)$$

where $\partial \mathbf{K}_{*, \mathbf{x}}, \partial^2 \mathbf{K}_{*,*}$ are matrices given by

$$(\partial \mathbf{K}_{\mathbf{x}, *})_{n,l} = \frac{\partial k(\mathbf{x}_n, \mathbf{x}_*)}{\partial x^{(l)}}, \quad \begin{matrix} n = 1, \dots, N \\ l = 1, \dots, q \end{matrix} \quad (18)$$

$$(\partial^2 \mathbf{K}_{*,*})_{i,l} = \frac{\partial^2 k(\mathbf{x}_*, \mathbf{x}_*)}{\partial x^{(i)} \partial x^{(l)}}. \quad \begin{matrix} i = 1, \dots, q \\ l = 1, \dots, q \end{matrix} \quad (19)$$

The GP-LVM model provides an explicit mapping from the latent space to the observed space. This mapping defines

the support of the observed data \mathbf{Y} as a q dimensional manifold embedded into \mathbb{R}^p . If the covariance function of the model is continuous and differentiable, the Jacobian of the GP-LVM mapping is well-defined and the natural metric follows Eq. 14.

It follows from Eq. 17 and the properties of the GPs that the distribution of the Jacobian of the GP-LVM mapping is the product of p independent Gaussian distributions (one for each dimension of the dataset) with mean $\boldsymbol{\mu}_{J(j,:)}$ and covariance $\boldsymbol{\Sigma}_J$. For a every latent point \mathbf{x}_* the Jacobian takes the following form:

$$\begin{aligned} p(\mathbf{J} \mid \mathbf{Y}, \mathbf{X}, \mathbf{x}_*) &= \prod_{j=1}^p \mathcal{N}(\boldsymbol{\mu}_{J(j,:)}, \boldsymbol{\Sigma}_J) \\ &= \prod_{j=1}^p \mathcal{N}(\partial \mathbf{K}_{\mathbf{x},*}^\top \tilde{\mathbf{K}}_{\mathbf{x},\mathbf{x}}^{-1} \mathbf{Y}_{:,j}, \partial^2 \mathbf{K}_{*,*} - \partial \mathbf{K}_{\mathbf{x},*}^\top \tilde{\mathbf{K}}_{\mathbf{x},\mathbf{x}}^{-1} \partial \mathbf{K}_{\mathbf{x},*}), \end{aligned} \quad (20)$$

which (c.f. Eq. 14) gives a distribution over the metric tensor \mathbf{G}

$$\mathbf{G} = \mathcal{W}_q(p, \boldsymbol{\Sigma}_J, \mathbb{E}[\mathbf{J}^\top] \mathbb{E}[\mathbf{J}]). \quad (21)$$

From this distribution, the expected metric tensor can be computed as

$$\mathbb{E}[\mathbf{J}^\top \mathbf{J}] = \mathbb{E}[\mathbf{J}^\top] \mathbb{E}[\mathbf{J}] + p \boldsymbol{\Sigma}_J. \quad (22)$$

Note that the expectation of the metric tensor includes a covariance term. This implies that the metric tensor expands as the uncertainty over the mapping increases. Hence, curve lengths also increases when going through uncertain regions, and as a consequence geodesics will tend to avoid these regions.

The metric tensor defines the local geometric properties of the GP-LVM model and it can be used as a tool to data exploration. One way to visualise the tensor metric is through the differential volume of the high dimensional parallelepiped spanned by GP-LVM; this, for a latent dimension $q = 2$ is known as magnification factor and it has been introduced by [Bishop et al., 1997] for generative topographic mapping (and self organising maps). Its explicit formulation for GP-LVM is given by

$$\text{MF} = \sqrt{\det(\mathbb{E}[\mathbf{J}^\top \mathbf{J}])}. \quad (23)$$

An example of the magnification factor is shown in Fig. 3.

4.3 COMPUTING GEODESICS

Given a latent space endowed with an expected Riemannian metric, we now consider how to compute geodesics (shortest paths) between given points. Once a geodesic is computed its length can be evaluated through numerical integration of Eq. 4.

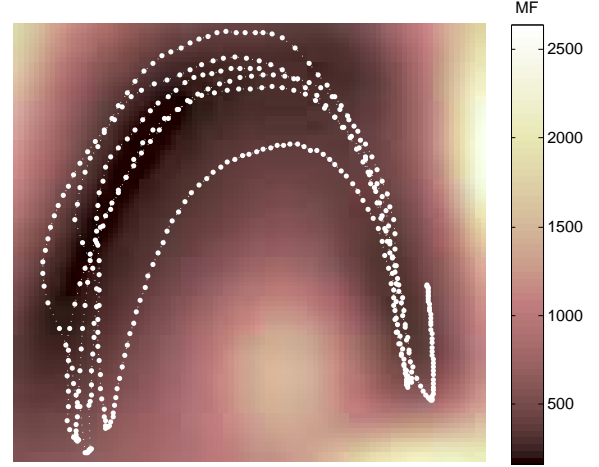


Figure 3: GP-LVM latent space for the motion capture data (see section 5 for details). White dots denote latent points \mathbf{x}_n and the background colour is proportional to the magnification factor (23).

The obvious solution to the shortest path problem is to discretise the latent space and compute shortest paths on the resulting graph using e.g. Dijkstra’s algorithm [Cormen et al., 1990]. The computational complexity of this approach, however, grows exponentially with the dimensionality of the latent space and the approach quickly becomes infeasible. Further, this approach will also introduce discretisation errors due to the finite size of the graph.

Instead we solve the geodesic differential equation (6) numerically. This scales more gracefully as it only involves a discretisation of the geodesic curve which is always one-dimensional independently of the dimension of the latent space. The 2nd order ODE in (6) can be rewritten in a standard way as a system of 1st order ODEs, which we can solve using a four-stage implicit Runge-Kutta method [Kierzenka and Shampine, 2001]². This gives a smooth solution which is fifth order accurate. Alternatively, such equations can be solved by repeated Gaussian process regression [Hennig and Hauberg, 2014].

To evaluate Eq. 6 we need the derivative of the expected metric:

$$\frac{\partial \text{vec} \mathbb{E}[\mathbf{G}(\mathbf{x})]}{\partial \mathbf{x}} = \frac{\partial \text{vec}(\mathbb{E}[\mathbf{J}^\top] \mathbb{E}[\mathbf{J}] + p \cdot \text{cov}(\mathbf{J}, \mathbf{J}))}{\partial \mathbf{x}}. \quad (24)$$

For the GP-LVM this reduces to computing the derivatives of the covariance function k . Given two vectors $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^q$, a widely used covariance function is the *squared exponential* (or *RBF*) kernel

$$k(\mathbf{x}_1, \mathbf{x}_2) = \alpha \exp\left(-\frac{\omega}{2} \|\mathbf{x}_1 - \mathbf{x}_2\|_2^2\right). \quad (25)$$

²We use an off-the-shelf numerical solver (bvp5c in Matlab[®]); runnig times and computational cost are provided in the reference.

We choose here the *RBF* as an illustrative example, but our approach apply to any other kernel that leads to a differential mapping. This function is differentiable in \mathbf{x} and will be used here (and in Section 5) to provide a specific algorithm. We explicitly compute Eq. 18 and 19 for the squared exponential kernel to have an explicit form of Eq. 20:

$$(\partial \mathbf{K}_{\mathbf{x}_1, \mathbf{x}_2})_{1,j} = -\omega(x_1^{(j)} - x_2^{(j)}) k(\mathbf{x}_1, \mathbf{x}_2) \quad (26)$$

$$\begin{aligned} (\partial^2 \mathbf{K}_{\mathbf{x}_1, \mathbf{x}_2})_{i,l} &= \\ &= \begin{cases} \omega(x_1^{(i)} - x_2^{(i)})(x_1^{(l)} - x_2^{(l)}) k(\mathbf{x}_1, \mathbf{x}_2), & i \neq l \\ \omega(\omega(x_1^{(i)} - x_2^{(i)})^2 - 1) k(\mathbf{x}_1, \mathbf{x}_2), & i = l \end{cases} \end{aligned} \quad (27)$$

Due to symmetry, the upper triangular of the Hessian matrix is sufficient to the computation. Note that, for our choice of kernel, the Hessian is diagonal and constant for $\mathbf{x}_1 = \mathbf{x}_2$, which is the case of $\partial^2 \mathbf{K}_{*,*}$, so there is no need to compute its derivative (which appears in the expression of $\partial \text{vec } \mathbf{G}$).

5 EXPERIMENTS AND RESULTS

Section 1 shows a first motivating example: a single image of a hand-written digit is rotated from 0 to 360 degrees to produce 200 rotated images. We then estimate³ a GP-LVM model with a $q = 2$ dimensional latent space; the latent space is shown in Fig. 1. We interpolate two points using either a straight line or a geodesic, and reconstruct images along these paths. The results in Fig. 4 show the poor reconstruction of the straight-line interpolator. The core problem with this interpolator is that it goes through regions with little data support, meaning the resulting reconstruction will be similar to the average of the entire data set.

In the next two sections we consider experiments on real data, but our results are similar to the synthetic digit experiment. First, we consider images of rotating objects (Section 5.1), and then motion capture data (Section 5.2).

5.1 IMAGES OF ROTATING OBJECTS

We consider images from the COIL data set [Nene et al., 1996], which consist of images from a fixed camera depicting 100 different objects on a motorised turntable against a black background. Each image is acquired after a 5 degree rotation of the turntable, giving a total of 72 images per object. Here we consider the images of object 74 (a rubber duck), but similar results are attained for other objects.

We estimate a $q = 2$ dimensional latent space using GP-LVM, and interpolate two latent points using either a straight line or a geodesic. Reconstructed images along the

³Software from the Machine Learning group, University of Sheffield <http://staffwww.dcs.shef.ac.uk/people/N.Lawrence/software.html>



Figure 4: Rotated digit. Inference after sampling over the latent space following the Geodesic distance (top row) and the Euclidean distance (bottom row); see also Fig. 1. Images are inverted and bicubically upsampled for improved viewing.

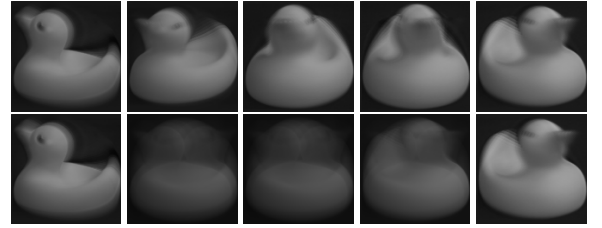


Figure 5: COIL image reconstruction. Inference after sampling over the latent space following the geodesic (top row) and the Euclidean straight line (bottom row).

interpolated paths are shown in Fig. 5. It is clear that the geodesic gives a better interpolation as it avoids regions with high uncertainty.

To measure the quality of the different interpolators we reconstruct 50 images equidistantly along each interpolating path and measure the distance to the nearest neighbour in the training data. This is shown in Fig. 6, which, for reference, also shows the average reconstruction error of the latent representations of the training data,

$$\text{Avg. training error} = \frac{1}{N} \sum_{n=1}^N \|\mathbb{E}[f(\mathbf{x}_n)] - \mathbf{y}_n\|. \quad (28)$$

It is clear that the straight line interpolator performs poorly away from the end-points, while the geodesic provides errors which are comparable to the average error of the latent representation of the training data.

5.2 HUMAN MOTION CAPTURE

We next consider human motion capture data from the *CMU Motion Capture Database*⁴. Specifically, we study motion 16 from subject 22, which is a repetitive *jumping jack* motion. Each time instance of this data consist of a human pose as acquired by a marker-based motion capture system; see Fig. 9 for example data. We represent each

⁴<http://mocap.cs.cmu.edu/>

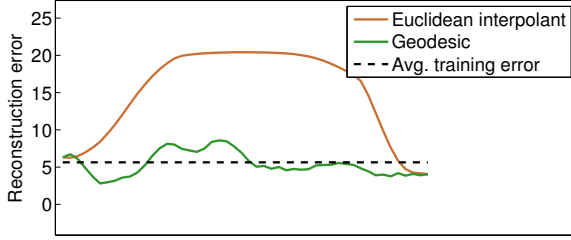


Figure 6: COIL reconstruction error. Inference after sampling over the latent space following the geodesic (green) and the Euclidean straight line (brown). For reference, the average reconstruction error of the latent observations is shown as well (dashed). This figure is best viewed in colour.

pose by the three-dimensional joint positions, i.e. as a vector $\mathbf{y}_{n,:} \in \mathbb{R}^{3P}$, where P denotes number of joint positions.

We estimate a GP-LVM using dynamics [Damianou et al., 2011] as is common for this type of data [Wang et al., 2008]. The resulting latent space is shown in Fig. 7, and the metric tensor is shown in Fig. 3. As can be seen, the latent points $\mathbf{x}_{n,:}$ follow a periodic pattern as expected for this motion, and the metric tensor is generally smaller in regions of high data density.

We pick two latent extremal points of the motion (\mathbf{x}_1 and \mathbf{x}_T) and interpolate them using the Euclidean straight line and the expected Riemannian geodesic. Fig. 7 show the interpolants: again, the geodesic follow the trend of the data while the straight line goes through regions with high model uncertainty. Reconstructed poses along the interpolants are shown in Fig. 10 and 11. A comparison with the intermediate poses ($\mathbf{x}_2 \dots \mathbf{x}_{T-1}$) in the training sequence (see Fig. 9) show that the geodesic interpolant is a more truthful reconstruction compared to that of the straight line.

To measure the quality of the reconstruction we note that the length of the subject’s limbs should stay constant throughout the sequence. Our representation does, however, not enforce this constraint. Fig. 8 show the length of the subjects forearm for the two reconstructions along with the correct length. The straight line interpolant drastically changes the limb lengths, while the geodesic matches the ground truth well. Similar observations have been made for other limbs.

6 DISCUSSION AND FUTURE WORK

When the mapping between a latent space and the observation space is not isometric (the common case for nonlinear mappings), a Euclidean distance measure in the latent space does not match that of the original observation space. In fact, the distance measures in the latent and observation spaces can be arbitrarily different. This makes it difficult to

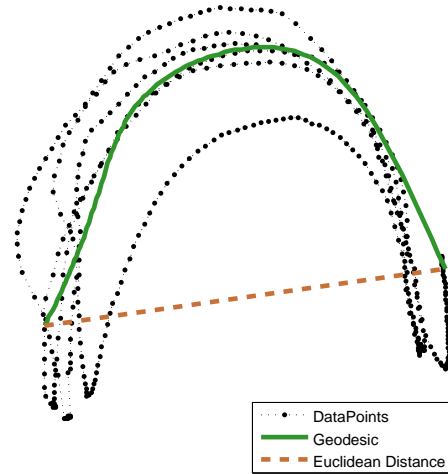


Figure 7: GP-LVM latent space for the motion capture data. White dots denote latent points \mathbf{x}_n and the background colour is proportional to the magnification factor (23). The blue curve denotes the geodesic interpolant, while the dashed red curve is the straight-line interpolant. This figure is best viewed in colour.

perform any meaningful statistical operation directly in the latent space as the used metric is difficult to interpret.

We solve this issue by carrying the metric from the observation space into the latent space in the form of a *random Riemannian metric*. This gives a distribution over a smoothly changing local metric at each point in the latent space. We then provide an expression for the *expected* local metric and show how shortest paths (geodesics) can be computed numerically under the resulting metric. These geodesics provide natural generalisations of straight-lines and are, thus, suitable for interpolation under the new metric.

For the GP-LVM model the expected metric depends on the uncertainty of the model, such that distances become longer in regions of high uncertainty. This effectively forces geodesic curves to avoid uncertain regions in the latent space, which is the desired behaviour for most applications. It is worth noting that a similar analysis for the GTM does *not* provide a metric with this capacity as the uncertainty is constant in this model.

The idea of considering the expected metric is practical as it turns the latent space into a Riemannian manifold. This opens up to many applications as statistical operations are reasonably well-understood in these spaces. E.g. tracking can be performed in the latent space through a Riemannian Kalman filter [Hauberg et al., 2013], classification can be done using the geodesic distance, etc.

It is, however, potentially misleading to only consider the expectation of the metric rather than the entire distributions

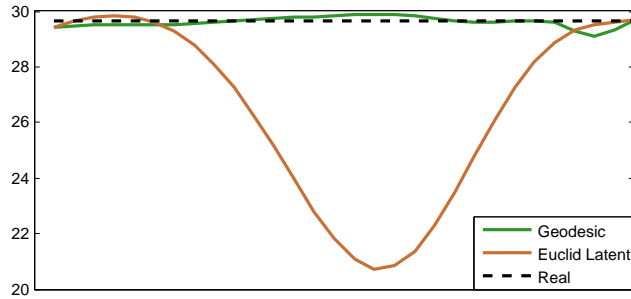


Figure 8: Length, in centimetres, of the subjects forearm during latent space interpolation. The blue curve is according to the geodesic interpolant, and the red dashed curve is according to the straight-line interpolant. For reference, the black dots show the true length.

of metrics. Although, if the latent dimension is much lower than the data dimension, it can be shown that the distribution of the metric concentrates around its mean. But in general *random Riemannian manifolds* are mathematically less well-understood, e.g. it is known that geodesics are almost surely not length minimising curves under a random metric [LaGatta and Wehr, 2014]. We are suggesting that manifolds derived from data are necessarily uncertain, and there is much to gain from further consideration of these spaces, which then naturally lead to distributions over geodesics, distances, angles, curvature and so forth.

In this paper we have only considered how geometry can be used to understand an already estimated LVM, but it is also worth considering if this geometry can be used when estimating the LVM. E.g. it is worth investigating if a prior on the curvature of the latent manifold is an effective way to influence learning.

Acknowledgements

The authors found great inspiration in the discussions at the *1st Braitenberg Round table on Probabilistic numerics and Random Geometries*. This research was partially funded by European research project EU FP7-ICT (Project Ref 612139 "WYSIWYD") and by Spanish research project TIN2012-31377. S.H. is funded in part by the Danish Council for Independent Research (Natural Sciences); the Villum Foundation; and an amazon.com machine learning in education award.

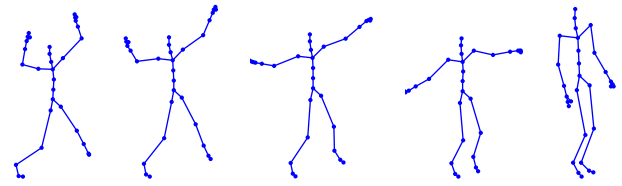


Figure 9: Example poses from the motion capture data. These poses are temporally between the end-points of the interpolating curves, i.e. they are comparable to the interpolated reconstructions.

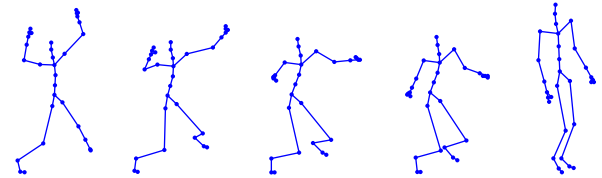


Figure 10: Interpolated poses according to the straight-line interpolant. In particular, note the bending of the knees, which does not occur in the training data.

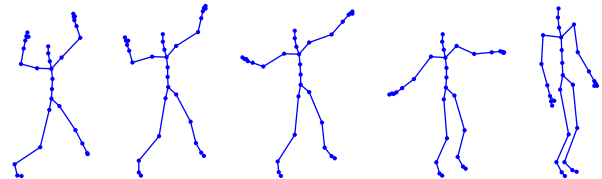


Figure 11: Interpolated poses according to the geodesic. These are visually similar to the poses in Fig. 9.

References

- S. Amari and H. Nagaoka. *Methods of information geometry*. Translations of mathematical monographs; v. 191. American Mathematical Society, 2000.
- T. W. Anderson. The non-central wishart distribution and certain problems of multivariate statistics. *The Annals of Mathematical Statistics*, 17(4):409–431, Dec. 1946.
- D. J. Bartholomew. *Latent Variable Models and Factor Analysis*. Charles Griffin & Co. Ltd, London, 1987.
- C. M. Bishop, M. Svensén, and C. K. I. Williams. Magnification factors for the SOM and GTM algorithms. In *Proceedings 1997 Workshop on Self-Organizing Maps, Helsinki University of Technology, Finland.*, pages 333–338, 1997.
- C. M. Bishop, M. Svensén, and C. K. I. Williams. GTM: the Generative Topographic Mapping. *Neural Computation*, 10(1):215–234, 1998. doi: 10.1162/089976698300017953.
- C. Bregler and S. M. Omohundro. Nonlinear image inter-

- polation using manifold learning. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *NIPS*, pages 973–980. MIT Press, 1994.
- T. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 1990.
- A. Damianou, M. K. Titsias, and N. D. Lawrence. Variational Gaussian process dynamical systems. In P. Bartlett, F. Peirerra, C. Williams, and J. Lafferty, editors, *Advances in Neural Information Processing Systems*, volume 24, Cambridge, MA, 2011. MIT Press.
- M. P. do Carmo. *Riemannian Geometry*. Birkhäuser Boston, January 1992.
- C. F. Gauss. Disquisitiones generales circa superficies curvas. *Commentationes Societatis Regiae Scientiarum Gottingensis Recentiores*, VI:99–146, 1827.
- A. Gonczarek and J. Tomczak. Manifold regularized particle filter for articulated human motion tracking. In J. Switek, A. Grzech, P. Switek, and J. M. Tomczak, editors, *Advances in Systems Science*, volume 240 of *Advances in Intelligent Systems and Computing*, pages 283–293. Springer International Publishing, 2014.
- K. Grochow, S. L. Martin, A. Hertzmann, and Z. Popović. Style-based inverse kinematics. *ACM Trans. Graph.*, 23(3):522–531, Aug. 2004.
- T. Hastie and R. Tibshirani. Discriminant adaptive nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(6):607–616, June 1996.
- S. Hauberg, O. Freifeld, and M. Black. A geometric take on metric learning. In P. Bartlett, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *Advances in Neural Information Processing Systems (NIPS) 25*, pages 2033–2041. MIT Press, 2012.
- S. Hauberg, F. Lauze, and K. S. Pedersen. Unscented kalman filtering on riemannian manifolds. *Journal of Mathematical Imaging and Vision*, 46(1):103–120, May 2013.
- P. Hennig and S. Hauberg. Probabilistic solutions to differential equations and their application to riemannian statistics. In *Proceedings of the 17th international Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 33, 2014.
- J. Kierzenka and L. F. Shampine. A BVP solver based on residual control and the Matlab PSE. *ACM Transactions on Mathematical Software*, 27(3):299–316, 2001.
- T. LaGatta and J. Wehr. Geodesics of random riemannian metrics. *Communications in Mathematical Physics*, 327(1):181–241, 2014.
- N. D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 11 2005.
- N. D. Lawrence. A unifying probabilistic perspective for spectral dimensionality reduction: Insights and new models. *Journal of Machine Learning Research*, 13, 2012. URL <http://jmlr.csail.mit.edu/papers/v13/lawrence12a.html>.
- J. Lee and M. Verleysen. Nonlinear dimensionality reduction. In *Information Science and Statistics*, Springer, 2007.
- S. A. Nene, S. K. Nayar, and H. Murase. Columbia object image library (coil-100). Technical Report CUCS-006-96, Department of Computer Science, Columbia University, Feb 1996.
- I. Olier and A. Vellido. Advances in clustering and visualization of time series using gtm through time. *Neural Networks*, 21(7):904–913, 2008.
- D. Ramanan and S. Baker. Local distance functions: A taxonomy, new algorithms, and an evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(4):794–806, 2011.
- C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006. ISBN 0-262-18253-X.
- J. Tenenbaum, V. Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.
- J. B. Tenenbaum. Mapping a manifold of perceptual observations. In M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, *NIPS*. The MIT Press, 1997. ISBN 0-262-10076-2.
- M. Tenenbaum and H. Pollard. *Ordinary Differential Equations*. Dover Publications, 1963.
- M. E. Tipping and C. M. Bishop. Mixtures of probabilistic principal component analysers. *Neural Computation*, 11(2):443–482, 1999.
- R. Urtasun, D. Fleet, A. Hertzmann, and P. Fua. Priors for people tracking from small training sets. In *International Conference on Computer Vision (ICCV)*, volume 1, pages 403–410, Oct 2005.
- A. Vellido, J. Martín, F. Rossi, and P. Lisboa. Seeing is believing: The importance of visualization in real-world machine learning applications. In *European Symposium on Artificial Neural Networks (ESANN)*, pages 219–226, 2011.
- J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Recognition and Machine Intelligence (PAMI)*, 30(2):283–298, Feb. 2008.

Efficient Regret Bounds for Online Bid Optimisation in Budget-Limited Sponsored Search Auctions

Long Tran-Thanh¹, Lampros Stavrogiannis¹, Victor Naroditskiy¹
Valentin Robu^{1,2}, Nicholas R Jennings¹ and Peter Key²

1: University of Southampton, UK
{l1t08r, ls8g09, vn, vr2, nrj}@ecs.soton.ac.uk

2: Microsoft Research Cambridge, UK
peter.key@microsoft.com

Abstract

We study the problem of an advertising agent who needs to intelligently distribute her budget across a sequence of online keyword bidding auctions. We assume the closing price of each auction is governed by the same unknown distribution, and study the problem of making provably optimal bidding decisions. Learning the distribution is done under censored observations, i.e. the closing price of an auction is revealed only if the bid we place is above it. We consider three algorithms, namely ε -First, Greedy Product-Limit (GPL) and LuekerLearn, respectively, and we show that these algorithms provably achieve Hannan-consistency. In particular, we show that the regret bound of ε -First is at most $O(T^{\frac{2}{3}})$ with high probability. For the other two algorithms, we first prove that, by using a censored data distribution estimator proposed by Zeng [19], the empirical distribution of the closing market price converges in probability to its true distribution with a $O(\frac{1}{\sqrt{t}})$ rate, where t is the number of updates. Based on this result, we prove that both GPL and LuekerLearn achieve $O(\sqrt{T})$ regret bound with high probability. This in fact provides an affirmative answer to the research question raised in [1]. We also evaluate the abovementioned algorithms using real bidding data, and show that although GPL achieves the best performance on average (up to 90% of the optimal solution), its long running time may limit its suitability in practice. By contrast, LuekerLearn and ε -First proposed in this paper achieve up to 85% of the optimal, but with an exponential reduction in computational complexity (a saving up to 95%, compared to GPL).

1 INTRODUCTION

Sponsored search is the most significant example of monetisation of Internet activities. This multi-billion dollar industry poses many challenging research problems for both advertisers and search engines. One of the most well-studied, but nonetheless still open, problems is the optimisation of marketing campaigns for an advertiser, or an autonomous agent acting on her behalf¹, with a fixed budget. This fundamental problem has been studied in a number of stylised models, yet many of the questions arising in real sponsored search auctions remain unanswered. In this paper, we focus on one such question—bidding when prices are not known but must be learnt to choose the right bidding strategy.

In this work, we follow a stochastic market price model that was used in [1, 7]. In particular, we take the point of view of an advertising agent with a specified budget for a given time horizon, who wants to find a bidding strategy that maximises the number of clicks. We consider a model with a single keyword and a single slot. Each time a user searches for the keyword, an auction is run to decide which of the interested agents is assigned the ad slot on the search results page. The winner is the agent with the highest bid who pays the *market price* which is determined by the second highest bid: i.e., the slot is sold in the style of a second-price auction. In practice, other factors affecting allocation of the slot include randomisation used by the search engine and advertiser/keyword-specific “quality scores” that adjust advertisers’ bids. Given these factors and the lack of information about bids and strategies of the other advertisers, an advertising agent cannot easily take into account her own effect on the market price, and so instead views the price as a random variable.

The key challenge of this stochastic model is that the distribution of the market price is not known in ad-

¹We will interchangeably use the terms agent and advertiser within this paper.

vance. Thus, to select the right bidding strategy, the agent needs to learn that distribution. This learning problem is further complicated by “censored observations” [1, 7]: the agent observes the market price only when she wins the auction; otherwise she just learns that the market price is above her bid. Although existing methods, designed for budget-limited online optimisation, can provably achieve asymptotically optimal performance for the case with no censorship [2, 18], they fail within the settings studied here. In particular, due to the censored observations, these methods cannot reproduce efficient estimation of the distribution of the market price, as they use conventional empirical estimation techniques [9, 16].

To combat this, Amin *et al.* (2012) proposed Greedy Product-Limit (GPL) and LuekerLearn, two methods that use the Kaplan–Meier estimator [9], designed for estimating the distribution of censored data, and achieve good performance in experiments with real bidding data. However, no theoretical performance analysis has been provided. Against this background, this paper addresses this gap by providing theoretical justification for these algorithms and a novel one that we develop for this setting. Our results prove asymptotic optimality of the algorithms, guaranteeing good performance as the number of auctions increases. We first look at ε –First, an algorithm inspired by a class of methods designed for multi-armed bandits [5, 18], tailored to our settings. In particular, this algorithm uses the first ε fraction of the auctions to estimate the market price distribution. Based on this estimate, it then solves a Markov decision process (MDP) in order to determine the optimal bidding policy. We prove that this algorithm achieves Hannan-consistency (i.e., sub-linear regret bound). Put differently, we show that the regret (i.e., the difference between the performance of a particular algorithm and that of an optimal solution) of the algorithm is at most $O(T^{\frac{2}{3}})$ with high probability, where T is the number of auctions. Note that the Hannan-consistency property (i.e., the sub-linear $O(T^{\frac{2}{3}})$ regret bound) guarantees that the average regret (i.e., the total regret divided by the number of auctions) converges to 0 as the number of auctions is increased, and thus, the bidding behaviour of a Hannan-consistent algorithm becomes more similar to that of the optimal solution (due to the decreasing performance gap defined by the average regret).

In addition to ε –First, we also provide an affirmative answer to the conjectures posed in [1]. That is, we show that, by replacing the Kaplan–Meier estimator with a novel censored data distribution estimator proposed by Zeng [19], GPL and LuekerLearn, the algorithms studied by Amin *et al.*, do indeed achieve sub-linear regret bounds, and thus, are also Hannan-consistent. In particular, we show that, by using Zeng’s estimator, the empirical distribution of the clos-

ing market price converges in probability to its true distribution with a $O(\frac{1}{\sqrt{t}})$ rate, where t is the number of updates. Relying on this result, we prove that GPL achieves $O(\sqrt{T})$ regret bound with high probability. On the other hand, LuekerLearn achieves $O(\sqrt{T} + \ln T)$ regret bound, also with high probability. Given this, our work extends the state of the art as follows:

- We provide a theoretical regret analysis for ε –First, GPL and LuekerLearn, and we show that they achieve Hannan-consistency.
- We compare the performance of each algorithm through extensive empirical evaluations, using real bidding data from Microsoft adCenter. In particular, we demonstrate that, although GPL typically outperforms the other algorithms, it requires significantly higher computational complexity, which could limit its suitability in practice. On the other hand, both ε –First and LuekerLearn can achieve performance close to that of GPL (typically within 10%), but with a much lower computational cost (with up to 50 times speed-up in computation time).

The remainder of the paper is organised as follows. In the next section we review related work. The model we study is presented in Section 3. We review existing and new algorithms for learning and bidding in Section 4. Our main contribution — theoretical guarantees — are derived in Section 5. Numerical evaluation using real-world data sets is presented in Section 6, and Section 7 concludes.

2 RELATED WORK

Bid optimisation in sponsored search auctions is a topic of considerable research in the autonomous agents community [4, 8, 10, 12, 14]. One of the first papers on the topic offers heuristic algorithms for prediction and bidding that were shown to work in practice [10]. Moreover, Feldman *et al.* [6] prove that simple randomised strategies for optimising a budget across multiple keywords achieve good performance. In that work, cost per click and number of clicks for each bid are known to the bidder. Berg *et al.* [3] compare bidding algorithms such as equating return-on-investment (ROI) and knapsack-based solutions based on the predictions they require (e.g., number of clicks and cost per click) and evaluate them in a simulated bidding environment of the Trading Agent Competition in Ad Auctions [8]. Unlike all the above papers, our focus is on bidding with online *learning*—in order to make bidding decisions, we need to learn the distribution of the market price.

The two papers closest to our work that combine learning and bidding in ad auctions are [1] and [20]. In par-

ticular, our work can be seen as a continuation of the research started by Amin *et al.* [1] who compared various algorithms for prediction and bidding. We adopt the same model, but focus on theoretical guarantees of the algorithms considered in [1] as well as that of our proposed ε -First algorithm. Zhou and Naroditskiy [20] address the keyword bidding problem when multiple slots are available, but do not provide theoretical guarantees for their proposed algorithm.

Furthermore, two very recent related works are [18] and [2]. Both works propose general frameworks for studying multi-armed bandit problems with supply (or budget) constraints. Although bidding in repeated auctions is a problem that can be modeled in these frameworks, they do not address the one-sided censored observations issue, which is the main challenge addressed here. It is worth noting that we can still apply these models to our settings by combining them with the censored data solutions described in Section 4. However, since they are designed for more generic problems, they do not exploit the domain-specific features of our problem, and thus, they provide weaker performance guarantees. Nevertheless, they may form a strong basis for our future work.

Finally, it is worth to note that our problem can also be formalised as a Markov decision process (MDP) (see Section 3 for more details), and thus, it shows similarities to the domain of reinforcement learning [15, 17]. However, as existing RL methods do not take into account censored data, it is not trivial how to incorporate them into our settings. Given this, we ignore the large literature of RL, as we argue that they are out of scope of our paper. Nevertheless, a possible future work would be to find an efficient way to combine RL techniques with censored data estimation.

3 MODEL DESCRIPTION

Our model consists of a sequence of T single slot second-price auctions, where the bidder (or agent) has to repeatedly place her bid in order to win a single keyword at each time step $t \in \{1, \dots, T\}$. We refer to T auctions as a bidding *period* and use B to denote the budget for the period. That is, the total cost spent on the auctions cannot exceed this budget. At each time step t , we assume that the market price x_t of the keyword is an independent and identically distributed (i.i.d.) random variable drawn from an unknown, but fixed, distribution with probability distribution function p . We assume that p has a finite support $[0, C]$ for some sufficiently large $C > 0$. This assumption is reasonable, as the market price is typically less than a couple of dollars. In our model, if a particular bid of the agent at time step t is higher than x_t , the agent wins the auction, and the budget is decreased by x_t . Otherwise, the agent does not win, and the budget

remains the same. More formally, let b_t and B_t denote the agent's bid, and the residual budget (i.e., the remaining budget) at time step t , respectively. Note that $B_1 = B$. Given this, we have

$$B_{t+1} = B_t - x_t$$

if $b_t \geq x_t$, and

$$B_{t+1} = B_t$$

otherwise. Note that the agent cannot place a bid that is higher than the current residual budget. That is, $b_t \leq B_t$ for each time step t . We assume that both b_t and x_t are discrete values chosen/drawn from \mathbb{Z}^+ . This assumption is reasonable, as the bids and market prices can be regarded as multiplications of the smallest unit of currency allowed for bidding.

Now, our goal is to find a bidding policy that maximises the number of wins over the time interval $\{1, \dots, T\}$. It is worth to note that if $B \geq CT$, we can achieve the optimal solution by repeatedly bidding with C . In particular, since bidding C always guarantees winning, if our budget is larger than CT , we can always win at each time step. Given this, we now only focus on the nontrivial case, and thus, we from hereafter assume that

$$B < CT \tag{1}$$

Given this condition, our problem can be formalised as follows. Let A denote a bidding policy that places bid $b^A(B_t, t)$ at each time step t , where B_t is the residual budget at that time step. In addition, let $G^A(B, T)$ denote the expected total number of wins of policy A with respect to total budget B and time limit T :

$$G^A(B, T) = \mathbb{E} \left[\sum_{t=1}^T I\{b^A(B_t, t) \geq x_t\} \right], \tag{2}$$

where $I\{\cdot\}$ is the indicator function. Note that $b^A(B_t, t) \leq B_t$ and

$$B_{t+1} = \begin{cases} B_t - x_t, & \text{if } b^A(B_t, t) \geq x_t \\ B_t, & \text{otherwise.} \end{cases}$$

We aim to find an optimal policy

$$A^* = \arg \max_A G^A(B, T)$$

that maximises the expected total number of wins. For the sake of simplicity, we denote the expected performance of A^* with $G^*(B, T)$. It is known that if we have exact information about the distribution function p , we can calculate A^* using a Markov decision process (MDP) formulation [1, 15]. In particular, let $F(b) = P(X > b)$ denote the survival function of the market price² (i.e., the probability that the market

²The problem of estimating the distribution of censored data first appeared in the survival analysis literature [9, 13]. Hence the name of the survival function.

price is higher than bid b). Suppose that the optimal policy A^* chooses bid $b^*(B', t)$ if the budget is B' at time step t . It can be shown that A^* has to satisfy the following set of Bellman equations [15]:

$$\begin{aligned}
b^*(B', t) &= \arg \max_{b(B', t)} \left\{ \sum_{\sigma=1}^{b(B', t)} p(\sigma) [1 + G^*(B' - \sigma, T - t)] \right. \\
&\quad \left. + F(b(B', t))G^*(B', T - t) \right\} \\
G^*(B', T - t + 1) &= \sum_{\sigma=1}^{b^*(B', t)} p(\sigma) [1 + G^*(B' - \sigma, T - t)] \\
&\quad + F(b^*(B', t))G^*(B', T - t)
\end{aligned}$$

for each $t \in \{1, \dots, T\}$ and $0 \leq B' \leq B$. That is, $b^*(B', t)$ denotes the optimal bid (i.e., the one that maximises the expected number of future wins) at time step t and budget B' , while the second equation implies that the optimal number of wins at time step t and budget B' can be achieved by taking the optimal bid and continuing with the optimal policy A^* (for more details, see e.g. [1, 15]). Note that $G^*(B', 0) = 0$ for any $0 \leq B' \leq B$. Given this, we can recursively solve the Bellman equations given above, and thus, determine the optimal bid for each time step t in order to calculate the optimal solution $G^*(B, T)$. Hereafter we may refer to A^* as the optimal stochastic solution, as opposed to the deterministic approach, that additionally has full information about the sequence of market prices x_t , which A^* typically does not have (see Section 4.3 for more details).

Since p is unknown for us, A^* cannot be determined in an exact manner. This implies that A^* represents a theoretical optimum value, which is unachievable in general. Nevertheless, for any algorithm A , we can define the regret for A as the difference between the total number of wins of A and that of the theoretical optimum A^* . More precisely, letting R^A denote the regret, we have

$$R^A(B, T) = G^*(B, T) - G^A(B, T)$$

Thus, our objective is to derive algorithms for learning p and bidding that minimise this regret.

4 ALGORITHMS

Given the problem definition, we now turn to the description of the algorithms that we study within this paper. In particular, we investigate three algorithms: (i) ε -First, (ii) GPL and (iii) LuekerLearn. These algorithms are described in the next sections.

4.1 The ε -First Algorithm

Algorithm 1 The ε -First Algorithm

```

1: Inputs:  $T > 0$ ,  $B > 0$ ,  $0 < \varepsilon < 1$ ;
2: Exploration phase:
3: for  $t = 1 \rightarrow \varepsilon T$  do
4:   randomly choose bid  $b_t$  from uniform distribution over  $[1, \frac{B}{\varepsilon T}]$ ;
5:   observe  $o_t = \min\{x_t, b_t\}$ ;
6: end for
7: Exploitation phase:
8: use Suzukawa's estimator to calculate  $\hat{p}$ ;
9: solve the Bellman equations given in Equation 4;
10: for  $t = \varepsilon T \rightarrow T$  do
11:   place the bid  $b^+(B_t, t)$  accordingly to the solution of the Bellman equations;
12: end for

```

As mentioned earlier, the key challenge of finding an optimal solution for the budget-limited auction problem is that we do not know the distribution function p of the market price in advance. Given this, we need to learn (or estimate) this distribution from the observed sequence of market prices x_1, x_2, \dots, x_T . This naturally lends itself to the idea of ε -First, which first estimates the distribution of the market price and then optimises the bidding policy. In particular, it uses an ε fraction of the total number of auctions T within a period to estimate the market price distribution function p . Following this, in the rest of $(1 - \varepsilon)T$ auctions, we solve the budget-limited auction problem with the estimated market price distribution function \hat{p} learnt from the learning phase. Hereafter we refer to the former phase as *exploration*, while to the latter as *exploitation*, respectively. In what follows, we describe these phases in more detail (the pseudo code is depicted in Algorithm 1).

We start with the description of the exploration phase. Within this phase, our goal is to accurately estimate the market price distribution. To do so, we can use the first ε proportion of the total auctions T . Now, recall that we can only observe x_t when it is not higher than the chosen bid b_t . That is, the sequence of x_t is (right) censored by the sequence of b_t . In particular, at each time step, we can only observe the value of $o_t = \min\{x_t, b_t\}$. This, indeed, makes the estimation of p a challenging problem. Note that [1] used the product-limit, or Kaplan-Meier (KM), estimator to address this challenge [9]. However, it is well known that the KM estimator has a negative bias [13]. To overcome this issue, we consider a modification of the KM estimator, an estimation technique proposed by [16], for estimating functionals of the distribution p . This method is proven to be unbiased, and thus, we can use McDiarmid's inequality to guarantee the $O(t^{-1})$ convergence rate of the \hat{p}_t estimate. This convergence rate provides the basis for the performance analysis of ε -First (see Section 5 for more details).

Suzukawa's method can be adopted to the estimation

of the market price distribution as follows. It relies on the assumption that we know the distribution from which the bids b_t are drawn. Let S denote the survival function of this bid distribution, and $o_t = \min\{x_t, b_t\}$ denote the observed value at t . Let $\varphi_b(x)$ be a function defined as

$$\begin{aligned}\varphi_b(x) &= 1 & \text{if } x \leq b \\ \varphi_b(x) &= 0 & \text{otherwise.}\end{aligned}$$

In addition, let $\delta_t = I\{x_t \leq b_t\}$ denote the indicator function whether the market price does not exceed the bid at time step t . Given this, Suzukawa's estimation for the market price's cumulative probability function P is formalised as:

$$\hat{P}_t(b) = \frac{1}{t} \sum_{i=1}^t \frac{\delta_i \varphi_b(o_i)}{S^-(o_i)} \quad (3)$$

where $S^-(o_i) = \lim_{x>0, x \rightarrow 0} S(o_i - x)$ and $\hat{P}_t(b)$ is the estimate of $P(b)$ after t observations. Using techniques similar to those from [16], we can easily derive that $\hat{P}_t(b)$ is indeed an unbiased estimator of $P(b)$.

Based on this, ε -First places the bids within the exploration phase as follows. For each $t \leq \varepsilon T$, ε -First uniformly chooses a bid b_t from $[1, \frac{B}{\varepsilon T}]$ (Algorithm 1, lines 4 – 5). This guarantees that the total cost spent within the exploration phase will not exceed the total budget B . When the exploration ends, let \hat{p} and \hat{F} denote Suzukawa's KM estimation of the market price distribution function p , and the survival function, respectively (line 8). Next, we will describe how ε -First uses these estimates to tackle the budget-limited auction problem.

We now turn to the description of the exploitation phase. Let $B_{\varepsilon T}$ denote the residual budget after the exploration phase ends. In order to determine the bids at each time step, ε -First solves the following Bellman equations:

$$\begin{aligned}b^+(B', t) &= \arg \max_{b(B', t)} \left\{ \sum_{\sigma=1}^{b(B', t)} \hat{p}(\sigma) [1 + G^+(B' - \sigma, T - t)] \right. \\ &\quad \left. + \hat{F}(b(B', t)) G^+(B', T - t) \right\} \\ G^+(B', T - t + 1) &= \sum_{\sigma=1}^{b^+(B', t)} \hat{p}(\sigma) [1 + G^+(B' - \sigma, T - t)] \\ &\quad + \hat{F}(b^+(B', t)) G^+(B', T - t) \quad (4)\end{aligned}$$

for each $\varepsilon T \leq t \leq T$ and $0 \leq B' \leq B_{\varepsilon T}$, where $b^+(B', t)$ is the chosen bid of ε -First at time step t and budget B' . Recall that $G^+(B', 0) = 0$ for any $0 \leq B' \leq B$. These together allow us to (recursively)

Algorithm 2 The GPL Algorithm

```

1: Inputs:  $T > 0, B > 0, \hat{p}_1$  is uniform;
2: for  $t = 1 \rightarrow T$  do
3:   solve the Bellman equations given in Equation 5
   for  $\hat{p}_t$ ;
4:   place a bid  $b^+(B_t, t)$  according to the solution of
   the Bellman equations;
5:   use Zeng's estimator to update  $\hat{p}_{t+1}$ ;
6: end for

```

evaluate each value of $b^+(B', t)$, and thus, the bidding policy within the exploitation phase of ε -First (Algorithm 1, lines 9 – 12).

The intuition behind ε -First is that by properly setting the value of ε , we can quickly estimate the distribution of the market price with sufficient accuracy. Thus, the solution of the Bellman equations within the exploitation phase is close to the optimal solution, resulting in a good overall bidding performance (see Section 5 for more details).

4.2 The GPL Algorithm

The GPL algorithm, introduced by [1], can be described as follows. For each time step t , it uses an MDP model to determine the current optimal policy, given the current estimate \hat{p}_t of the market price distribution function p . That is, it solves a set of Bellman equations, similar to the exploitation phase of ε -First, but with a different \hat{p}_t at each time step. According to this optimal policy, it then chooses the next bid, and observes the censored value o_t . Based on this observation, GPL uses a novel censored data distribution estimator, proposed by [19], to update the estimation of the market price distribution function, \hat{p}_{t+1} , for the next time step. Note that here we replace the KM estimator, which is used in [1], with Zeng's method (for a brief description of Zeng's method and further explanations, see Section 5). The above mentioned steps are repeated until $t = T$ (see Algorithm 2 for the pseudo code). More formally, suppose that the residual budget at time step t is B_t . In addition, let \hat{F}_t denote the estimate of the survival function at t . GPL solves the following equations:

$$b^+(B', \tau) = \arg \max_{b(B', \tau)} \left\{ \sum_{\sigma=1}^{b(B', \tau)} \hat{p}_t(\sigma) [1 + G^+(B' - \sigma, T - \tau)] + \hat{F}_t(b(B', \tau)) G^+(B', T - \tau) \right\}$$

$$\begin{aligned}G^+(B', T - \tau) &= \sum_{\sigma=1}^{b^+(B', \tau)} \hat{p}_t(\sigma) [1 + G^+(B' - \sigma, T - \tau - 1)] \\ &\quad + \hat{F}_t(b^+(B', \tau)) G^+(B', T - \tau - 1)\end{aligned}$$

where $t \leq \tau \leq T-1$ and $0 \leq B' \leq B_t$. In addition, we have $G^+(B', 0) = 0$ for all $0 \leq B' \leq B_t$. For the sake of simplicity, we set \hat{p}_1 to be a uniform distribution in $(0, B]$. Given the solutions, GPL then places bid $b^+(B_t, t)$ at each time step t (Algorithm 2, lines 2–6).

4.3 The LuekerLearn Algorithm

Similar to GPL, this algorithm was also described in [1], and is based on the algorithm proposed by Lueker for the online stochastic knapsack problem [11]. In particular, within the online stochastic knapsack problem, an item with profit r_t and weight x_t arrives into the system at each time step t such that the pair $\{r_t, x_t\}$ is drawn from a *fixed* and *known* joint distribution. At each time step, we have to decide whether to put the arrived item into a knapsack with the total capacity B such that the total weight of the chosen items cannot exceed this capacity. Our goal is to maximise the total profit of the chosen items. It is easy to see that within our settings, if the market price distribution p is known in advance, the budget-limited auction problem can be reduced to the online stochastic knapsack problem by setting $r_t = 1$ for each t . Given this, Lueker's algorithm, originally designed for the online stochastic knapsack problem, can be adopted to the budget-limited auction with full knowledge of p as follows (for more details, see [11]). At each time step $1 \leq t \leq T$, Lueker's algorithm chooses a bid $b^+(B_t, t)$ that satisfies

$$b^+(B_t, t) = \max \{b\} \quad \text{s.t.} \quad \sum_{\sigma=0}^b p(\sigma)\sigma \leq \frac{B_t}{T-t+1} \quad (5)$$

where B_t is the current residual budget. The efficiency of this algorithm is guaranteed by the following:

Proposition 1 (Theorem 2 from [11]) *Suppose that we have full information about the market price distribution p . Consider the optimal deterministic solution, that has the full information about the sequence of market prices $\{x_t\}$ as well (i.e., it knows the value of each x_t in advance). Given this, the difference between the performance of Lueker's algorithm and that of the optimal deterministic solution is at most $O(\ln T)$.*

The proof can be found in [11]. However, since neither the sequence of $\{x_t\}$ nor p is known in advance, we combine Lueker's algorithm with Zeng's estimator (instead of the KM estimator) in order to learn the market price distribution and determine an efficient bid at the same time. This leads to the LuekerLearn algorithm (see Algorithm 3), that places a bid $b^+(B_t, t)$ as follows:

Algorithm 3 The LuekerLearn Algorithm

```

1: Inputs:  $T > 0$ ,  $B > 0$ ,  $\hat{p}_1$  is uniform;
2: for  $t = 1 \rightarrow T$  do
3:   place a bid  $b^+(B_t, t)$  according to Equation 6;
4:   use Zeng's estimator to update  $\hat{p}_{t+1}$ ;
5: end for

```

$$b^+(B_t, t) = \max \{b\} \quad \text{s.t.} \quad \sum_{\sigma}^b \hat{p}_t(\sigma)\sigma \leq \frac{B_t}{T-t+1} \quad (6)$$

where \hat{p}_t is the estimate of p at time step $1 \leq t \leq T$, and $b^+(B_T, T) = B_T$. Based on the censored observation o_t , it then updates the estimation of p (i.e., \hat{p}_{t+1}), using the Zeng's estimator (Algorithm 3, lines 2–5). The intuition of the algorithm is that as the estimate \hat{p}_t gets more accurate over time, the algorithm converges to the original algorithm provided by Lueker. Since Proposition 1 guarantees the efficiency of the latter, LuekerLearn can also achieve low regret bounds, as we will prove later in this work.

5 PERFORMANCE ANALYSIS

Within this section, we analyse the performance of the aforementioned algorithms. In particular, we derive performance regret bounds for each of the algorithms. We also show that these regret bounds imply the fact that the algorithms converge to the theoretical optimal solution with high probability. We start with ε -First:

Theorem 2 *Let $T \geq 8(-\ln \frac{\beta}{2})$ for some $0 < \beta < 1$. For any $0 < \varepsilon < 1$ and $B > \varepsilon CT$, and $T > C$ where C is the support of the market price, the regret of the modified version of ε -First where Suzukawa's method is used for the estimation of the market price distribution, is at most $C\varepsilon T + \sqrt{\frac{8(-\ln \frac{\beta}{2})T}{\varepsilon}}$ with probability of at least $(1-\beta)$. In addition, by setting $\varepsilon = \left(\frac{-2\ln \frac{\beta}{2}}{C^2 T}\right)^{\frac{1}{3}}$, the regret bound can be refined to $3\left(-2\ln \frac{\beta}{2}\right)^{\frac{1}{3}} C^{\frac{1}{3}} T^{\frac{2}{3}}$.*

Note that the condition $B > \varepsilon CT$ guarantees that within the exploration phase, the bids are uniformly sampled from the entire interval $[1, C]$, since the algorithm samples from the $[1, \frac{B}{\varepsilon T}]$. This condition guarantees that Suzukawa's estimator can fully cover the interval $[1, C]$. In addition, the $O(C^{\frac{1}{3}} T^{\frac{2}{3}})$ regret bound is weak if the $C < T$ condition does not hold. In particular, by fixing T and increasing C , we will get a regret bound that is worse than $O(T)$. Nevertheless, this regret bound achieves Hannan consistency (i.e., sub-linear in T) if $C < T$.

It is also worth to note that since we only consider the case $B \sim O(T)$, $O(T^{\frac{2}{3}})$ regret bound is equivalent

lent to $O(B^{\frac{2}{3}})$, as $T^{\frac{2}{3}} > \frac{B^{\frac{2}{3}}}{C^{\frac{2}{3}}}$. Given the results for ε -First, we now turn to the analysis of GPL and LuekerLearn. If we consider the sequence of the bids as random variables, then the consistency and the zero bias property of KM estimators such as Suzukawa's require independency between the bids and the market price³. However, since in both GPL and LuekerLearn we choose the current bid based on the empirical distribution which is built by using the previous observations, it is easy to see that the current bid is not independent from the sequence of the previous market prices. Thus, the consistency (and the convergence rate) of the empirical distribution might not be guaranteed if the standard KM or Suzukawa's estimator is used within GPL and LuekerLearn (for more details, see, e.g., [13, 16]). This implies that neither GPL or LuekerLearn can achieve Hannan-consistency if we use their versions proposed in [1] without any modifications. To overcome this issue, we replace the KM estimator within GPL and LuekerLearn with a novel censored data estimator proposed by Zeng [19]. Due to its complexity and the space limitations, the detailed description of Zeng's estimator is omitted (for more details, see [19]). However, we sketch it as follows.

Zeng's method assumes that there is an underlying set of (known) variables \mathbf{L} that describes the dependency between the two sequences of chosen bids and market prices. Furthermore, suppose that \mathbf{L} is sufficient enough such that for each t , x_t (i.e., the market price) is independent from b_t (i.e., the chosen bid value), conditional to the value of \mathbf{L} at time step t , denoted with \mathbf{L}_t . In addition, this method requires that either x_t or b_t follows Cox's proportional model; that is, at least one of the following conditions must hold:

$$p(x_t | \mathbf{L}_t = \mathbf{l}) \sim \lambda_x \exp \{ \beta' \mathbf{l} \} \quad (7)$$

$$p(b_t | \mathbf{L}_t = \mathbf{l}) \sim \lambda_b \exp \{ \gamma' \mathbf{l} \} \quad (8)$$

for some unknown λ_x, λ_b random variables, and some (unknown) parameters β and γ , respectively. Let \hat{P}_t denote Zeng's estimate of the market price P after t time steps. Zeng proved that $\sqrt{t}(\hat{P}_t - P)$ is a Donsker-class empirical process. Based on this result, we state the following:

Theorem 3 *The abovementioned assumptions hold for both GPL and LuekerLearn. Given this, by using Zeng's estimation method, the estimate \hat{P}_t of the market price distribution converges in probability to the true distribution P with rate $O(\frac{1}{\sqrt{t}})$ in both GPL and LuekerLearn.*

This theorem implies the following statements:

³In fact, it is sufficient to guarantee that the covariance between the bids and the market price is 0.

Theorem 4 *There exists a constant $K > 0$ that only depends on the market price distribution p , such that the regret of GPL, combined with Zeng's estimator, is at most $O(2K\sqrt{T})$ with high probability.*

Similarly, we have the following theorem for LuekerLearn:

Theorem 5 *There exists a constant $K > 0$ that only depends on the market price distribution p , such that the regret of LuekerLearn, combined with Zeng's estimator, is at most $O(2K(\sqrt{T} + \ln T))$ with high probability.*

Similarly to the case of ε -First, here we can also transform the regret bounds of GPL and LuekerLearn to $O(2K\sqrt{B})$ and $O(2K(\sqrt{B} + \ln B))$, respectively.

Note that Theorems 4 and 5 imply that GPL converges faster to the optimal solution than LuekerLearn, as T tends to infinity. This is due to the additional $\ln T$ term within the regret bound of LuekerLearn. The reason behind this is that LuekerLearn in fact converges with rate $O(\ln T)$ towards GPL. Hence an additional, $O(\ln T)$, gap is needed here. Also note that by using Zeng's method in ε -First, we would get worse results, compared to Theorem 2, as with the approach from Suzukawa, we could derive exact constant coefficient values for the regret bound, while Zeng's method only provides asymptotic regret bounds. In addition, since in both Theorems 4 and 5, the value of K is typically hard to be identified, the results of these theorems are in fact focussing on the asymptotic behaviour of the algorithms (i.e., both algorithms are Hannan consistent), and do not address whether the bounds are tight.

6 NUMERICAL EVALUATION

While we have so far developed theoretical upper bounds for the performance regret of the algorithms, we now turn to practical aspects and examine their performance in a realistic setting, as it might be the case that regret bound for ε -First is not tight, and thus, it might perform better than $O(T^{\frac{2}{3}})$ in many cases, as we will demonstrate later within this section. Given this, in this section, we aim to investigate whether the algorithms achieve high performance when applied to practical sponsored search auction problems. To do so, we first describe our parameter settings in Section 6.1. We then continue with the numerical results of the algorithms' performance in Section 6.2.

6.1 Parameter Settings

To investigate the performance of the algorithms, we use the same dataset as [1], taken from a real-world

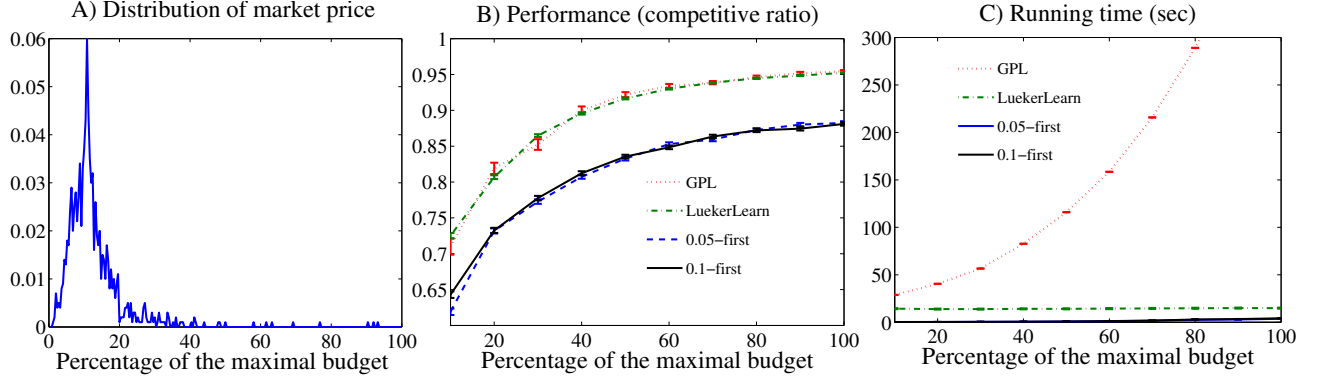


Figure 1: Numerical results on a subset of keywords with single distribution peaks, and with budgets ranging from 10% to 100% of the maximal budget $B_k(T)$: A) A typical single-peaked distribution of the market price. B) The performance of the algorithms, measured in competitive ratio against the optimal solution. C) Computational cost of the algorithms.

sponsored search auction database. Given this, we follow the parameter settings described there. In particular, for each experiment, we use $U = 10$ periods, each of which comprises $T = 100$ auctions and the budget is refilled at the beginning of each period. For a fair comparison to the results of Amin *et al.*, the maximal budget $B_k(T)$ for keyword k is also selected in the same way they do. In particular, we set $B_k(T)$ such that $G^*(B_k(T), T) = fT$ for $f = 0.1$ (i.e., 10%) and $T = 100$. This setting aims to satisfy that, on average, we can win 10% of the auctions. Within each experiment, we vary the budget from $\frac{B_k(T)}{10}$ up to $B_k(T)$ with a step of $\frac{B_k(T)}{10}$. Each experiment was repeated 100 times (for more details of the parameter settings, see [1]). Within our experiments, we run ε -First with $\varepsilon = 0.05$ and $\varepsilon = 0.1$, respectively, as these values are typically more efficient than other value settings⁴.

6.2 Numerical Results

Given the description of the parameter settings above, we now investigate the numerical results in more detail. In particular, we observed that the real distribution of the market price can typically be distinguished into two groups. In the first group, the market price usually concentrates at low values, creating a single-peaked distribution (see Figure 1A). Within the second group, the market price is typically more scattered, causing multiple peaks within the distribution (see Figure 2A). The performance efficiency of the algorithms also vary between these distribution groups. Therefore, we distinguish these two cases, and separately examine the performance of the algorithms

within these cases. In particular, Figure 1 depicts the numerical results for the single-peaked case, and Figure 2 depicts the results for the multi-peaked case, respectively (here, the second group typically contains two peaks, as is also shown in Figure 2).

We first evaluate the single-peaked case (Figure 1). As mentioned earlier, Figure 1A shows the distribution of the market price. In addition, Figure 1B plots the performance of the algorithm, compared against that of the optimal stochastic solution described in Section 3. Here, the optimal stochastic solution also uses an MDP model to determine the optimal bidding policy, but assuming full knowledge of the distribution of market prices. Figure 1C depicts the running time of each algorithm. As can be seen from the figures, GPL and LuekerLearn provide similar performance, and both outperform the two versions of ε -First, 0.05-First and 0.1-First, by up to 10%. The reason for this is that since the market price is typically concentrated at low values, all the algorithms can quickly learn this. This allows GPL and LuekerLearn to use small bids to refine the estimation of the market price distribution at small values, and thus, to bid more efficiently. In contrast, as ε -First stops learning after the exploration phase, its estimation at the small values is not as accurate as the others'. Given this, ε -First bids suboptimally in more time steps, compared to the other two. Nevertheless, note that ε -First can still achieve by up to 88% of the optimal solution.

On the other hand, the running time of GPL is significantly larger, compared to that of the others (Figure 1C). In particular, GPL typically needs more than 500 seconds to evaluate the case of maximal budget $B_k(T)$, while ε -First algorithms only need less than 10 seconds. The reason for this is that GPL recomputes the MDP for the optimal decision at each step, after updating its price distribution. This is

⁴Note that all the numerical tests appearing in this paper are performed on a personal computer, Intel® Xeon® CPU W3520 @2.67GHz with 12GB RAM and under Windows 7 operating system. The code was written and tested on Matlab R2012a.

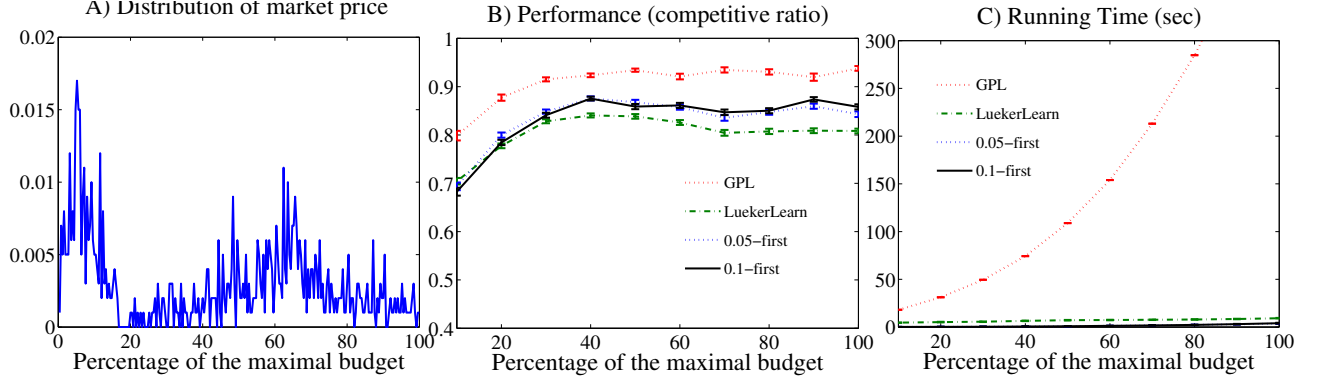


Figure 2: Numerical results on a subset of keywords with more than one distribution peaks, and with budgets ranging from 10% to 100% of the maximal budget $B_k(T)$: A) A typical two-peaked distribution of the market price. B) The performance of the algorithms, measured in competitive ratio against the optimal solution. C) Computational cost of the algorithms.

computationally expensive, especially for large budgets. By contrast, the ε -First algorithms only compute the MDP once, at the end of the exploration phase. Thus, despite its best competitive ratio performance, the running time of GPL would limit its suitability for real-time deployment. LuekerLearn also needs approximately 15 seconds to solve this problem instance. Given this, for single-peaked distributions, LuekerLearn yields the best trade-off between efficiency and computational cost, as it achieves similar performance to that of the GPL (33 times faster), and is almost as fast as the ε -First algorithms.

Within the case of distributions with multiple peaks (in this case, we consider the two-peaked version), we can see that GPL still provides the best performance (see Figure 2B). However, in this setting, ε -First outperforms LuekerLearn by approximately 5%. The reason behind this is that due to multiple peaks, LuekerLearn starts to deviate between the peaks, as it makes more observations (see [1] for more details). This implies that LuekerLearn makes more suboptimal bids, as placing bids at the first peak is typically more desirable, as opposed to the bids close to the second peak. On the other hand, due to its restricted learning phase, ε -First typically learns the values around the first peak, and thus, can act more efficiently, compared to LuekerLearn. Nevertheless, both ε -First and LuekerLearn still achieve good performance, as both typically provide at least 80% of the optimal solution's.

In terms of computational cost, GPL still requires the highest running time (more than 600 seconds for the case of maximal budget $B_k(T)$). By contrast, both ε -First and LuekerLearn require at most 10 seconds. Note that ε -First is typically two times faster than LuekerLearn. Therefore, in the two-peaked case, ε -First is clearly the best choice for the budget-limited

auction problem, as it provides good performance (above 85% of the optimal solution), and achieves by far the lowest computational cost.

7 CONCLUSIONS

We studied the online bid optimisation problem in budget-limited sponsored search auctions, where the market price is drawn from a fixed, but unknown distribution, and is censored by the value of our current bid. Although existing algorithms have been shown to achieve good performance in practice, no theoretical performance analysis has been provided for this problem. Given this, we proposed ε -First, and we show that it provably achieves $O(T^{\frac{2}{3}})$ regret bound with high probability, where T is the number of total auctions. We also provided an affirmative answer to the research question raised in [1], which conjectures that GPL, a state-of-the-art algorithm for the budget-limited sponsored search auction problem, can achieve asymptotically optimal performance. In particular, we proved that GPL achieves $O(\sqrt{T})$ regret bound with high probability. We also showed in the paper that the regret bound of LuekerLearn, another state-of-the-art algorithm, is $O(\sqrt{T} + \ln T)$, also with high probability. In addition, we compared the performance of the algorithms on real-world data, and observed that, although GPL provides the highest performance, it is by far the most computationally expensive algorithm, and its running time would make it infeasible for real time deployment. On the other hand, LuekerLearn would be the best choice in the case of single-peaked distributions, as it provides the best trade-off between efficiency and computational cost. For the two-peaked distribution case, we showed that ε -First outperforms LuekerLearn with a reduced running time.

References

- [1] Amin, K., Kearns, M., Key, P., and Schwaighofer, A. (2012). Budget optimization for sponsored search: Censored learning in MDPs. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, UAI'12, pages 54–63.
- [2] Badanidiyuru, A., Kleinberg, R., and Slivkins, A. (2013). Bandits with knapsacks. In *IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 207–216.
- [3] Berg, J., Greenwald, A., Naroditskiy, V., and Sodomka, E. (2010). A first approach to autonomous bidding in ad auctions. In *Workshop on Trading Agent Design and Analysis at the 11th ACM Conference on Electronics Commerce*.
- [4] Engel, Y. and Chickering, D. M. (2008). Incorporating user utility into sponsored-search auctions. *Proceedings of the Seventh International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 1565–1569.
- [5] Even-Dar, E., Mannor, S., and Mansour, Y. (2002). PAC bounds for multi-armed bandit and Markov decision processes. In *COLT*.
- [6] Feldman, J., Muthukrishnan, S., Pal, M., and Stein, C. (2007). Budget optimization in search-based advertising auctions. In *Proceedings of the 8th ACM conference on Electronic commerce*, EC '07, pages 40–49. ACM.
- [7] Gummadi, R., Key, P., and Proutiere, A. (2012). Optimal bidding strategies and equilibria in dynamic auctions with budget constraints. Available at SSRN: <http://ssrn.com/abstract=2066175>.
- [8] Jordan, P. R., Wellman, M. P., and Balakrishnan, G. (2010). Strategy and mechanism lessons from the first ad auctions trading agent competition. In *Proceedings of the 11th ACM conference on Electronic commerce*, EC '10, pages 287–296, New York, NY, USA. ACM.
- [9] Kaplan, E. L. and Meier, P. (1958). Non-parametric estimation from incomplete observations. *Journal of the American Statistical Society*, **53**, 457–481.
- [10] Kitts, B. and Leblanc, B. (2004). Optimal bidding on keyword auctions. *Electronic Markets*, **14**(3), 186–201.
- [11] Lueker, G. S. (1995). Average-case analysis of off-line and on-line knapsack problems. In *Proceedings of the sixth annual ACM-SIAM symposium on Discrete algorithms*, SODA '95, pages 179–188, Philadelphia, PA, USA. Society for Industrial and Applied Mathematics.
- [12] Pardoe, D. and Stone, P. (2011). A particle filter for bid estimation in ad auctions with periodic ranking observations. *Proceedings of the Tenth International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 687–694.
- [13] Phadia, E. and Van Ryzin, J. (1980). A note on convergence rates for the product limit estimator. *The Annals of Statistics*, **8**(3), 673–678.
- [14] Stavrogiannis, L. C., Gerding, E. H., and Polukarov, M. (2013). Competing intermediary auctions. *Proceedings of the Twelfth International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 667–674.
- [15] Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. MIT Press.
- [16] Suzukawa, A. (2004). Unbiased estimation of functionals under random censorship. *Journal of the Japan Statistical Society*, **32**(2), 153–172.
- [17] Szepesvári, C. (2010). *Algorithms for Reinforcement Learning*. Morgan & Claypool Publishers.
- [18] Tran-Thanh, L., Chapman, A., Rogers, A., and Jennings, N. R. (2012). Knapsack based optimal policies for budget-limited multi-armed bandits. *Proceedings of the 26th Conference on Artificial Intelligence (AAAI 2012)*, pages 1134–1140.
- [19] Zeng, D. (2004). Estimating marginal survival function by adjusting for dependent censoring using many covariates. In *The Annals of Statistics*, pages 1533–1555.
- [20] Zhou, Y. and Naroditskiy, V. (2008). Algorithm for stochastic multiple-choice knapsack problem and keywords bidding. In *WWW08: Workshop on Targeting and Ranking for Online Advertising*.

A Consistent Estimator of the Expected Gradient Outerproduct

Shubhendu Trivedi*
TTI-Chicago

Jialei Wang*
University of Chicago

Samory Kpotufe
TTI-Chicago

Gregory Shakhnarovich
TTI-Chicago

Abstract

In high-dimensional classification or regression problems, the expected gradient outerproduct (EGOP) of the unknown regression function f , namely $\mathbb{E}_X (\nabla f(X) \cdot \nabla f(X)^\top)$, is known to recover those directions $v \in \mathbb{R}^d$ most relevant to predicting the output Y .

However, just as in gradient estimation, optimal estimators of the EGOP can be expensive in practice. We show that a simple rough estimator, much cheaper in practice, suffices to obtain significant improvements on real-world nonparametric classification and regression tasks. Furthermore, we prove that, despite its simplicity, this rough estimator remains statistically consistent under mild conditions.

1 INTRODUCTION

In high-dimensional nonparametric classification or regression problems, the output Y might not depend equally on all input variables in $X = (X^i)_{i=1}^d$. To be more precise, let $Y \approx f(X)$ for some unknown smooth f , it is often the case that f varies most along a few *relevant* coordinates, and varies little along most coordinates. This observation has given rise to many practical variable selection methods.

The usual assumption in variable selection is that $f(X) = g(PX)$, where $P \in \{0, 1\}^{k \times d}$ projects X down to $k < d$ relevant coordinates. This assumption is generalized in *multi-index* regression (see e.g. [7, 9, 2, 12]) by letting $P \in \mathbb{R}^{k \times d}$ project X down to a k -dimensional subspace of \mathbb{R}^d . In other words, while f might vary significantly along all coordinates of X , it actually only depends on an unknown k -dimensional subspace.

Recovering this relevant subspace (sometimes called *effective dimension reduction* [7]) gives rise to the expected gra-

dient outerproduct (EGOP):

$$\mathbb{E}_X G(X) \triangleq \mathbb{E}_X (\nabla f(X) \cdot \nabla f(X)^\top).$$

The EGOP recovers the average variation of f in all directions: the directional derivative at x along $v \in \mathbb{R}^d$ is given by $f'_v(x) = \nabla f(x)^\top v$, in other words $\mathbb{E}_X |f'_v(X)|^2 = \mathbb{E}_X (v^\top G(X) v) = v^\top (\mathbb{E}_X G(X)) v$.

It follows that, if f does not vary along v , v must be in the null-space of the EGOP matrix $\mathbb{E}_X G(X)$, since $\mathbb{E}_X |f'_v(X)|^2 = 0$. In fact, it is not hard to show that, under mild conditions (f continuously differentiable on a compact space \mathcal{X}), the column space of $\mathbb{E}_X G(X)$ is exactly the *relevant* subspace defined by P ([11]).

Interestingly, the EGOP is useful beyond the above multi-index motivation: even if there is no clearly relevant dimension-reduction P , as is likely in practice, one can expect that f does not vary equally in all directions. Instead of dimension-reduction, we might rather weight any direction $v \in \mathbb{R}^d$ according to its relevance as captured by the average variation of f along v (encoded in the EGOP). The weighting approach will be the main use of EGOP considered in this work.

The EGOP can be estimated in various sophisticated ways, which can however be prohibitively expensive. For instance an optimal way of estimating $\nabla f(x)$, and hence the EGOP, is to estimate the slope of a linear approximation to f locally at each $x = X_i$ in an n -sample $\{(X_i, Y_i)\}_{i=1}^n$. Local linear fits can however be prohibitively expensive since it involves multiplying and inverting large-dimensional matrices at all X_i . This can render the approach impractical although it is otherwise well motivated.

The main message of this work is that the EGOP need not be estimated optimally, but just well enough to use towards improving classification or regression, our practical end-goal.

The cheaper estimator considered here is as follows. Let f_n denote an initial estimate of f (we use a kernel estimate);

*Both authors contributed equally to this work

for the i -th coordinate of $\nabla f(x)$, we use the rough estimate

$$\Delta_{t,i} f_n(x) = (f_n(x + te_i) - f_n(x - te_i))/2t, \quad t > 0.$$

Let $G_n(x)$ be the outer-product of the resulting gradient estimate $\hat{\nabla} f_n(x)$, the EGOP is estimated as $\mathbb{E}_n G_n(X)$, the empirical average of G_n . The exact procedure is given in Section 3.1.

We first show that this estimator is sound: despite being a rough approximation, it remains a statistically consistent estimate of the EGOP under very general distributional conditions. The main consistency result and key difficulties (having to do with interdependencies in the estimate) are discussed in Section 4.

More importantly, we show through extensive experiments that preprocessing data with this cheaper EGOP estimate can significantly improve the performance of nonparametric classification and regression procedures in real-world applications. This is described in Section 5.

In the next Section 2, we start with an overview of relevant work, followed by Section 3 describing the estimator and our theoretical setup.

2 SUMMARY OF RELEVANT WORK

The recent work of [6] considers estimating the coordinates f'_i of ∇f in a similar fashion as in the present work. However [6] is only concerned with a variable selection setting where each coordinate i of X is to be weighted by an estimate of $\mathbb{E}_X |f'_i(X)|$, which is their quantity of interest. This work addresses the more general approach of estimating the EGOP, its consistency and applicability.

Multiple methods have been developed for multi-index regression analysis, some using the so-called *inverse regression* approach (e.g. [7]), and many of them incorporating the estimation of derivative functionals of the unknown f . These approaches can already be found in early work such as [9], and typically estimate ∇f as the slope of local linear approximations of f .

Recent works of [11, 8] draw a clearer link between the various approaches to multi-index regression, and in particular relate the EGOP to the *covariance*-type matrices estimated in inverse regression. Furthermore, [8] proposes an alternative to estimating local linear slopes: their method estimates ∇f via a regularized least-squares objective over an RKHS. This is however still expensive since the least-square solution involves inverting an $n \times n$ feature matrix. In contrast our less sophisticated approach will take time in the order of n times the time to estimate f_n (f_n in practice could be a fast kernel regressor employing fast range-search methods).

The main use of the EGOP in the context of multi-index regression (as in the above cited work) is to recover the

relevant subspace given by P in the model $f(x) = g(Px)$. The data can then be projected to the estimated subspace before projection.

While we do not argue for a particular way to use the EGOP to preprocess data, our experiments focus on the following use: let VDV^\top be a spectral decomposition of the estimated EGOP, transform the input x as $D^{1/2}V^\top x$. Thus we do not rely on the multi-index model holding, but rather on a more general model where P might be a full-dimensional rotation (i.e. all directions are relevant), but g varies more in some coordinate than in others. The diagonal element $D_{i,i}$ recovers $\mathbb{E}_X (g'_i(X))^2$ where g'_i denotes coordinate i of ∇g , while V^\top recovers P .

3 SETUP AND DEFINITIONS

We consider a regression or classification setting where the input X belongs to a space $\mathcal{X} \subset \mathbb{R}^d$, of bounded diameter 1. The output Y is real. We are interested in the unknown *regression function* $f(x) \triangleq \mathbb{E}[Y|X = x]$ (in the case of classification with $Y \in \{0, 1\}$, this is just the probability of 1 given x).

For a vector $x \in \mathbb{R}^d$, let $\|x\|$ denote the Euclidean norm, while for a matrix A , let $\|A\|_2$ denote the spectral norm, i.e. the largest singular value $\sigma_{\max}(A)$.

We use $A \circ B$ to denote the entry-wise product of matrices A and B .

3.1 ESTIMATING THE EGOP

We let μ denote the marginal of $P_{X,Y}$ on \mathcal{X} and we let μ_n denote its empirical counterpart on a random sample $\mathbf{X} = \{X_i\}_{i=1}^n$. Given a labeled sample $(\mathbf{X}, \mathbf{Y}) = \{(X_i, Y_i)\}_{i=1}^n$ from $P_{X,Y}^n$, we estimate the EGOP as follows.

We consider a simple kernel estimator defined below, using a Kernel K satisfying the following admissibility conditions:

Definition 1 (Admissible Kernel). $K : \mathbb{R}_+ \mapsto \mathbb{R}_+$ is *non-increasing*, $K > 0$ on $[0, 1)$, and $K(1) = 0$.

Using such an admissible kernel K , and a bandwidth $h > 0$, we consider the regression estimate $f_{n,h}(x) = \sum_i \omega_i(x) Y_i$ where

$$\omega_i(x) = \frac{K(\|x - X_i\|/h)}{\sum_j K(\|x - X_j\|/h)} \text{ if } B(x, h) \cap \mathbf{X} \neq \emptyset,$$

$$\omega_i(x) = \frac{1}{n} \text{ otherwise.}$$

For any dimension $i \in [d]$, and $t > 0$, we first define

$$\Delta_{t,i} f_{n,h}(x) \triangleq \frac{f_{n,h}(x + te_i) - f_{n,h}(x - te_i)}{2t}.$$

This is a rough estimate of the line-derivative along coordinate i . However, for a robust estimate we also need to ensure that enough sample points contribute to the estimate. To this end, given a confidence parameter $0 < \delta < 1$ (this definition for δ is assumed in the rest of this work), define $A_{n,i}(X)$ as the event that

$$\min_{s \in \{-t, -t\}} \mu_n(B(X + se_i, h/2)) \geq \frac{2d \ln 2n + \ln(4/\delta)}{n}.$$

The gradient estimate is then given by the vector

$$\hat{\nabla} f_{n,h}(x) = (\Delta_{t,i} f_{n,h}(x) \cdot \mathbf{1}_{A_{n,i}(x)})_{i \in [d]}.$$

Note that, in practice we can just replace $A_{n,i}(X)$ with the event that the balls $B(X + se_i, h)$, $s \in \{-t, t\}$, contain samples.

Finally, define $G_n(x)$ as the outer-product of $\hat{\nabla} f_{n,h}(x)$, we estimate $\mathbb{E}_X G(X)$ as

$$\mathbb{E}_n G_n(X) \triangleq \frac{1}{n} \sum_{i=1}^n \hat{\nabla} f_{n,h}(X_i) \cdot \hat{\nabla} f_{n,h}(X_i)^\top.$$

3.2 DISTRIBUTIONAL QUANTITIES AND ASSUMPTIONS

For the analysis, our assumptions are quite general. In fact we could simply assume, as is common, that μ has lower-bounded density on a compact support \mathcal{X} , and that f is continuously differentiable; all the assumptions below will then hold. We list these more general detailed assumptions to better understand the minimal distributional requirements for consistency of our EGOP estimator.

A1 (Noise). Let $\eta(X) \triangleq Y - f(X)$. We assume the following general noise model: $\forall \delta > 0$ there exists $c > 0$ such that $\sup_{x \in X} \mathbb{P}_{Y|X=x}(|\eta(x)| > c) \leq \delta$. We denote by $C_Y(\delta)$ the infimum over all such c . For instance, suppose $\eta(X)$ has exponentially decreasing tail, then $\forall \delta > 0$, $C_Y(\delta) \leq O(\ln 1/\delta)$.

Last the variance of $(Y|X = x)$ is upper-bounded by a constant σ_Y^2 uniformly over $x \in X$. The next assumption is standard for nonparametric regression/classification.

A2 (Bounded Gradient). Define the τ -envelope of \mathcal{X} as $\mathcal{X} + B(0, \tau) \triangleq \{z \in B(x, \tau), x \in \mathcal{X}\}$. We assume there exists τ such that f is continuously differentiable on the τ -envelope $\mathcal{X} + B(0, \tau)$. Furthermore, for all $x \in \mathcal{X} + B(0, \tau)$, we have $\|\nabla f(x)\| \leq R$ for some $R > 0$, and ∇f is uniformly continuous on $\mathcal{X} + B(0, \tau)$ (this is automatically the case if the support \mathcal{X} is compact).

The next assumption generalizes common smoothness assumptions: it is typically required for gradient estimation that the gradient itself be Hölder continuous (or that f be

second-order smooth). These usual assumptions imply the more general assumptions below.

A3 (Modulus of continuity of ∇f). Let $\epsilon_{t,i} = \sup_{x \in \mathcal{X}, s \in [-t, t]} |f'_i(x) - f'_i(x + se_i)|$. We assume $\epsilon_{t,i} \xrightarrow{t \rightarrow 0} 0$ which is for instance the case when ∇f is uniformly continuous on an envelope $\mathcal{X} + B(0, \tau)$.

The next two assumptions capture some needed regularity conditions on the marginal μ . To enable local approximations of $\nabla f(x)$ over \mathcal{X} , the marginal μ should not concentrate on the boundary of \mathcal{X} . This is captured in the following assumption.

A4 (Boundary of \mathcal{X}). Define the (t, i) -boundary of \mathcal{X} as $\partial_{t,i}(\mathcal{X}) = \{x : \{x + te_i, x - te_i\} \not\subseteq \mathcal{X}\}$. Define the vector $\mu_{\partial_t} = (\mu(\partial_{t,i}(\mathcal{X})))_{i \in [d]}$. We assume that $\mu_{\partial_t} \xrightarrow{t \rightarrow 0} 0$. This is for instance the case if μ has a continuous density on \mathcal{X} .

Finally we assume that μ has mass everywhere, so that for samples X in dense regions, $X \pm te_i$ is also likely to be in a dense region.

A5 (Full-dimensionality of μ). For all $x \in \mathcal{X}$ and $h > 0$, we have $\mu(B(x, h)) \geq C_\mu h^d$. This is for instance the case if μ has a lower-bounded density on \mathcal{X} .

4 CONSISTENCY OF THE ESTIMATOR

$\mathbb{E}_n G_n(X)$ OF $\mathbb{E}_X G(X)$

We establish consistency by bounding $\|\mathbb{E}_n G_n(X) - \mathbb{E}_X G(X)\|_2$ for finite sample size n . The main technical difficulties in establishing the main result below have to do with the fact that each gradient approximation $\Delta_{t,h} f_{n,h}(X)$ at a sample point X depends on all other samples in \mathbf{X} . These inter-dependencies are circumvented by proceeding in steps which consider related quantities that are less sample-dependent.

Theorem 1 (Main). Assume A1, A2 and A5. Let $t < \tau$ and suppose $h \geq (\log^2(n/\delta)/n)^{1/d}$. There exist $C = C(\mu, K(\cdot))$ and $N = N(\mu)$ such that the following holds with probability at least $1 - 2\delta$. Define $A(n) = \sqrt{C d \cdot \log(n/\delta)} \cdot C_Y^2(\delta/2n) \cdot \sigma_Y^2 / \log^2(n/\delta)$. Suppose $n \geq N$, we have:

$$\begin{aligned} \|\mathbb{E}_n G_n(X) - \mathbb{E}_X G(X)\|_2 &\leq \frac{6R^2}{\sqrt{n}} \left(\sqrt{\ln d} + \sqrt{\ln \frac{1}{\delta}} \right) + \\ &\quad \left(3R + \|\epsilon_t\| + \sqrt{d} \left(\frac{hR + C_Y(\delta/n)}{t} \right) \right) \cdot \left[\|\epsilon_t\| + \right. \\ &\quad \left. \frac{\sqrt{d}}{t} \sqrt{\frac{A(n)}{nh^d}} + 2h^2 R^2 + R \left(\sqrt{\frac{d \ln \frac{d}{2n}}{2n}} + \|\mu_{\partial_t}\| \right) \right] \end{aligned}$$

Proof. Start with the decomposition

$$\begin{aligned} \|\mathbb{E}_n G_n(X) - \mathbb{E}_X G(X)\|_2 &\leq \|\mathbb{E}_n G(X) - \mathbb{E}_X G(X)\|_2 \\ &\quad + \|\mathbb{E}_n G_n(X) - \mathbb{E}_n G(X)\|_2. \end{aligned} \quad (1)$$

The two terms of the r.h.s. are bounded separately in Lemma 2 and 12. \square

Remark. Under the additional assumptions A3 and A4, the theorem implies consistency for $t \xrightarrow{n \rightarrow \infty} 0$, $h \xrightarrow{n \rightarrow \infty} 0$, $h/t^2 \xrightarrow{n \rightarrow \infty} 0$, and $(n/\log n)h^d t^4 \xrightarrow{n \rightarrow \infty} \infty$, this is satisfied for many settings, for example $t \propto h^{1/4}$, $h \propto (1/n)^{1/(2(d+1))}$.

The bound on the first term of (1) is a direct result of the below concentration bound for random matrices:

Lemma 1. [10, 3]. Consider a random matrix $A \in \mathbb{R}^{d \times d}$ with bounded spectral norm $\|A\|_2 \leq M$. Let A_1, A_2, \dots, A_n be i.i.d. copies of A . With probability at least $1 - \delta$, we have

$$\left\| \frac{1}{n} \sum_{i=1}^n A_i - \mathbb{E}A \right\|_2 \leq \frac{6M}{\sqrt{n}} \left(\sqrt{\ln d} + \sqrt{\ln \frac{1}{\delta}} \right).$$

We apply the above concentration to the i.i.d. matrices $G(X), X \in \mathbf{X}$, using the fact that $\|G(X)\|_2 = \|\nabla f(X)\|^2 \leq R^2$.

Lemma 2. Assume A2. With probability at least $1 - \delta$ over the i.i.d sample $\mathbf{X} \triangleq \{X_i\}_{i=1}^n$, we have

$$\|\mathbb{E}_n G(X) - \mathbb{E}_X G(X)\|_2 \leq \frac{6R^2}{\sqrt{n}} \left(\sqrt{\ln d} + \sqrt{\ln \frac{1}{\delta}} \right).$$

The next Lemma provides an initial bound on the second term of (1).

Lemma 3. Fix the sample (\mathbf{X}, \mathbf{Y}) . We have:

$$\begin{aligned} \|\mathbb{E}_n G_n(X) - \mathbb{E}_n G(X)\|_2 &\leq \mathbb{E}_n \|\nabla f(X) - \hat{\nabla} f_{n,h}(X)\| \\ &\quad \cdot \max_{x \in \mathbf{X}} \|\nabla f(x) + \hat{\nabla} f_{n,h}(x)\|. \end{aligned} \quad (2)$$

Proof. We have by a triangle inequality $\|\mathbb{E}_n G_n(X) - \mathbb{E}_n G(X)\|_2$ is bounded by:

$$\mathbb{E}_n \left\| \left(\hat{\nabla} f_{n,h}(X) \cdot \hat{\nabla} f_{n,h}(X)^\top - \nabla f(X) \cdot \nabla f(X)^\top \right) \right\|_2.$$

To bound the r.h.s above, we use the fact that, for vectors a, b , we have

$$aa^\top - bb^\top = \frac{1}{2}(a-b)(b+a)^\top + \frac{1}{2}(b+a)(a-b)^\top,$$

implying that

$$\begin{aligned} \|aa^\top - bb^\top\|_2 &\leq \frac{1}{2} \|(a-b)(b+a)^\top\|_2 \\ &\quad + \frac{1}{2} \|(b+a)(a-b)^\top\|_2 \\ &= \|(b+a)(a-b)^\top\|_2 \end{aligned}$$

since the spectral norm is invariant under matrix transposition.

We therefore have that $\|\mathbb{E}_n G_n(X) - \mathbb{E}_n G(X)\|_2$ is at most

$$\begin{aligned} \mathbb{E}_n \|(\nabla f(X) - \hat{\nabla} f_{n,h}(X)) \cdot (\nabla f(X) + \hat{\nabla} f_{n,h}(X))^\top\|_2 \\ = \mathbb{E}_n \|\nabla f(X) - \hat{\nabla} f_{n,h}(X)\| \cdot \|\nabla f(X) + \hat{\nabla} f_{n,h}(X)\| \\ \leq \mathbb{E}_n \|\nabla f(X) - \hat{\nabla} f_{n,h}(X)\| \cdot \max_{x \in \mathbf{X}} \|\nabla f(x) + \hat{\nabla} f_{n,h}(x)\|. \end{aligned}$$

\square

Thus the matrix estimation problem is reduced to that of an average gradient estimation. The two terms of (2) are bounded in the following two subsections. These sections thus contain the bulk of the analysis. All omitted proofs are found in the supplementary.

4.1 BOUND ON $\mathbb{E}_n \|\nabla f(X) - \hat{\nabla} f_{n,h}(X)\|$

The analysis of this section relies on a series of approximations. In particular we relate the vector $\hat{\nabla} f_{n,h}(x)$ to the vector

$$\hat{\nabla} f(x) \triangleq (\Delta_{t,i} f(x) \cdot \mathbf{1}_{A_{n,i}(x)})_{i \in [d]}.$$

In other words we start with the decomposition:

$$\begin{aligned} \mathbb{E}_n \|\nabla f(X) - \hat{\nabla} f_{n,h}(X)\| &\leq \mathbb{E}_n \|\nabla f(X) - \hat{\nabla} f(X)\| \\ &\quad + \mathbb{E}_n \|\hat{\nabla} f(X) - \hat{\nabla} f_{n,h}(X)\|. \end{aligned} \quad (3)$$

We bound each term separately in the following subsections.

4.1.1 Bounding $\mathbb{E}_n \|\nabla f(X) - \hat{\nabla} f(X)\|$

We need to introduce vectors $\mathbf{I}_n(x) \triangleq (\mathbf{1}_{A_{n,i}(x)})_{i \in [d]}$, and $\overline{\mathbf{I}_n(x)} \triangleq (\mathbf{1}_{\bar{A}_{n,d}(x)})_{i \in [d]}$. We then have:

$$\begin{aligned} \mathbb{E}_n \|\nabla f(X) - \hat{\nabla} f(X)\| &\leq \mathbb{E}_n \|\nabla f(X) \circ \overline{\mathbf{I}_n(X)}\| \\ &\quad + \mathbb{E}_n \|\nabla f(X) \circ \mathbf{I}_n(X) - \hat{\nabla} f(X)\|. \end{aligned} \quad (4)$$

The following lemma bounds the first term of (4).

Lemma 4. Assume A2 and A5. Suppose $h \geq (\log^2(n/\delta)/n)^{1/d}$. With probability at least $1 - \delta$ over the

sample of \mathbf{X} :

$$\mathbb{E}_n \left\| \nabla f(X) \circ \overline{\mathbf{I}_n(X)} \right\| \leq R \cdot \left(\sqrt{\frac{d \ln \frac{d}{\delta}}{2n}} + \|\mu_{\partial_t}\| \right).$$

The second term of (4) is bounded in the next lemma.

Lemma 5. *Fix the sample \mathbf{X} . We have $\max_{X \in \mathbf{X}} \|\nabla f(X) \circ \mathbf{I}_n(X) - \hat{\nabla} f(X)\| \leq \|\epsilon_t\|$.*

The last two lemmas can then be combined using equation (4) into the final bound of this subsection.

Lemma 6. *Assume A2 and A5. Suppose $h \geq (\log^2(n/\delta)/n)^{1/d}$. With probability at least $1 - \delta$ over the sample \mathbf{X} :*

$$\mathbb{E}_n \|\nabla f(X) - \hat{\nabla} f(X)\| \leq R \cdot \left(\sqrt{\frac{d \ln \frac{d}{\delta}}{2n}} + \|\mu_{\partial_t}\| \right) + \|\epsilon_t\|.$$

4.1.2 Bounding $\mathbb{E}_n \|\hat{\nabla} f(X) - \hat{\nabla} f_{n,h}(X)\|$

We need to consider bias and variance functionals of estimates $f_{n,h}(x)$. To this end we introduce the expected estimate $\tilde{f}_{n,h}(x) = \mathbb{E}_{\mathbf{Y}|\mathbf{X}} f_{n,h}(x) = \sum_{i=1}^n w_i(x) f(X_i)$.

The following lemma bounds the bias of estimates $f_{n,h}$. The proof relies on standard ideas.

Lemma 7 (Bias of $f_{n,h}$). *Assume A2. Let $t < \tau$. We have for all $X \in \mathbf{X}$, all $i \in [d]$, and $s \in \{-t, t\}$:*

$$|\tilde{f}_{n,h}(X + se_i) - f(X + se_i)| \cdot \mathbf{1}_{A_{n,i}(X)} \leq hR.$$

The following lemma bounds the variance of estimates $f_{n,h}$ averaged over the sample \mathbf{X} . To obtain a high probability bound, we rely on results of Lemma 7 in [6]. However in [6], the variance of the estimator is evaluated at a point, therefore requiring local density assumptions. The present lemma has no such local density requirements given that we are interested in an average quantity over a collection of points.

Lemma 8 (Average Variance). *Assume A1. There exist $C = C(\mu, K(\cdot))$, such that the following holds with probability at least $1 - 2\delta$ over the choice of the sample (\mathbf{X}, \mathbf{Y}) . Define $A(n) = \sqrt{Cd \cdot \ln(n/\delta)} \cdot C_Y^2(\delta/2n) \cdot \sigma_Y^2$, for all $i \in [d]$, and all $s \in \{-t, t\}$:*

$$\mathbb{E}_n |\tilde{f}_{n,h}(X + se_i) - f_{n,h}(X + se_i)|^2 \cdot \mathbf{1}_{A_{n,i}(X)} \leq \frac{A(n)}{nh^d}$$

The main bound of this subsection is given in the next lemma which combines the above bias and variance results.

Lemma 9. *Assume A1 and A2. There exist $C = C(\mu, K(\cdot))$, such that the following holds with probability at least $1 - 2\delta$ over the choice of (\mathbf{X}, \mathbf{Y}) . Define $A(n) = \sqrt{Cd \cdot \ln(n/\delta)} \cdot C_Y^2(\delta/2n) \cdot \sigma_Y^2$:*

$$\mathbb{E}_n \|\hat{\nabla} f(X) - \hat{\nabla} f_{n,h}(X)\| \leq \frac{\sqrt{d}}{t} \sqrt{\frac{A(n)}{nh^d}} + 2R^2h^2.$$

Proof. In what follows, we first apply Jensen's inequality, and the fact that $(a + b)^2 \leq 2a^2 + 2b^2$. We have:

$$\begin{aligned} \mathbb{E}_n \|\hat{\nabla} f(X) - \hat{\nabla} f_{n,h}(X)\| &= \mathbb{E}_n \left(\sum_{i \in [d]} |\Delta_{t,i} f_{n,h}(X) - \Delta_{t,i} f(X)|^2 \cdot \mathbf{1}_{A_{n,i}(X)} \right)^{1/2} \\ &\leq \left(\sum_{i \in [d]} \mathbb{E}_n |\Delta_{t,i} f_{n,h}(X) - \Delta_{t,i} f(X)|^2 \cdot \mathbf{1}_{A_{n,i}(X)} \right)^{1/2} \\ &\leq \frac{\sqrt{d}}{2t} \left(\max_{i \in [d], s \in \{-t, t\}} 4\mathbb{E}_n |f_{n,h}(\tilde{X}) - f(\tilde{X})|^2 \cdot \mathbf{1}_{A_{n,i}(X)} \right)^{1/2} \end{aligned} \quad (5)$$

where $\tilde{X} = X + se_i$. Next, use the fact that for any $s \in \{-t, t\}$, we have the following decomposition into variance and bias terms

$$\begin{aligned} &|f_{n,h}(X + se_i) - f(X + se_i)|^2 \\ &\leq 2|f_{n,h}(X + se_i) - \tilde{f}_{n,h}(X + se_i)|^2 \\ &\quad + 2|\tilde{f}_{n,h}(X + se_i) - f(X + se_i)|^2. \end{aligned}$$

Combine this into (5) to get a bound in terms of the average bias and variance of estimates $f_{n,h}(X + se_i)$. Apply Lemma 7 and 8 and conclude. \square

4.1.3 Main Result of this Section

The following theorem provides the final bound of this section on $\mathbb{E}_n \|\nabla f(X) - \hat{\nabla} f_{n,h}(X)\|$. It follows directly from the decomposition of equation 3 and Lemmas 6 and 9.

Lemma 10. *Assume A1, A2 and A5. Let $t < \tau$ and suppose $h \geq (\log^2(n/\delta)/n)^{1/d}$. With probability at least $1 - 2\delta$ over the choice of the sample (\mathbf{X}, \mathbf{Y}) , we have*

$$\begin{aligned} \mathbb{E}_n \|\nabla f(X) - \hat{\nabla} f_{n,h}(X)\| &\leq \frac{\sqrt{d}}{t} \sqrt{\frac{A(n)}{nh^d}} + 2R^2h^2 \\ &\quad + R \left(\sqrt{\frac{d \ln \frac{d}{\delta}}{2n}} + \|\mu_{\partial_t}\| \right) + \|\epsilon_t\|. \end{aligned}$$

4.2 BOUNDING $\max_{X \in \mathbf{X}} \|\nabla f(X) + \hat{\nabla} f_{n,h}(X)\|$

Lemma 11. Assume A1 and A2. With probability at least $1 - \delta$, we have

$$\|\nabla f(X) + \hat{\nabla} f_{n,h}(X)\| \leq 3R + \|\epsilon_t\| + \sqrt{d} \left(\frac{hR + C_Y(\delta/n)}{t} \right).$$

Proof. Fix $X \in \mathbf{X}$. We have

$$\begin{aligned} \|\nabla f(X) + \hat{\nabla} f_{n,h}(X)\| &\leq 2\|\nabla f(X)\| \\ &\quad + \|\nabla f(x) - \hat{\nabla} f_{n,h}(X)\| \\ &\leq 2R + \|\nabla f(X) - \hat{\nabla} f(x)\| \\ &\quad + \|\hat{\nabla} f(X) - \hat{\nabla} f_{n,h}(X)\|. \end{aligned} \tag{6}$$

We can bound the second term of (6) above as follows.

$$\begin{aligned} \|\nabla f(X) - \hat{\nabla} f(X)\| &\leq \|\nabla f(X) \circ \mathbf{I}_n(X) - \hat{\nabla} f(X)\| \\ &\quad + \|\nabla f(X) \circ \overline{\mathbf{I}_n(X)}\| \\ &\leq \|\epsilon_t\| + R, \end{aligned}$$

where we just applied Lemma 5.

For the third term of (6), $\|\hat{\nabla} f(x) - \hat{\nabla} f_{n,h}(x)\|$ equals

$$\sqrt{\sum_{i \in [d]} (|\Delta_{t,i} f_{n,h}(x) - \Delta_{t,i} f(x)| \cdot \mathbf{1}_{A_{n,i}(x)})^2}.$$

As in the proof of Lemma 9, we decompose the above summand into bias and variance terms, that is:

$$\begin{aligned} &|\Delta_{t,i} f_{n,h}(x) - \Delta_{t,i} f(x)| \\ &\leq \frac{1}{t} \max_{s \in \{-t, t\}} |\tilde{f}_{n,h}(x + se_i) - f(x + se_i)| \\ &\quad + \frac{1}{t} \max_{s \in \{-t, t\}} |\tilde{f}_{n,h}(x + se_i) - f_{n,h}(x + se_i)|. \end{aligned}$$

By Lemma 7, $|\tilde{f}_{n,h}(x + se_i) - f(x + se_i)| \leq Rh$ for any $s \in \{-t, t\}$.

Next, by definition of $C_Y(\delta/n)$, with probability at least $1 - \delta$, for each $j \in [n]$, Y_j has value within $C_Y(\delta)$ of $f(X_j)$. It follows that $|\tilde{f}_{n,h}(X + se_i) - f_{n,h}(X + se_i)| \leq C_Y(\delta/n)$ for $s \in \{-t, t\}$.

Thus, with probability at least $1 - \delta$, we have

$$\|\hat{\nabla} f(X) - \hat{\nabla} f_{n,h}(X)\| \leq \sqrt{d} \left(\frac{hR + C_Y(\delta/n)}{t} \right).$$

Combine these bounds in (6) and conclude. \square

4.3 FINAL BOUND ON $\|\mathbb{E}_n G_n(X) - \mathbb{E}_n G(X)\|_2$

We can now combine the results of the last two subsections, namely Lemma 10 and 11, into the next lemma, using the bound of Lemma 3.

Lemma 12. Assume A1, A2 and A5. Let $t < \tau$ and suppose $h \geq (\log^2(n/\delta)/n)^{1/d}$. With probability at least $1 - 2\delta$ over the choice of the sample (\mathbf{X}, \mathbf{Y}) , we have that $\|\mathbb{E}_n G_n(X) - \mathbb{E}_n G(X)\|_2$ is at most

$$\begin{aligned} &\left(3R + \|\epsilon_t\| + \sqrt{d} \left(\frac{hR + C_Y(\delta/n)}{t} \right) \right) \\ &\quad \left[\frac{\sqrt{d}}{t} \sqrt{\frac{A(n)}{nh^d} + 2h^2 R^2} + R \left(\sqrt{\frac{d \ln \frac{d}{\delta}}{2n}} + \|\mu_{\partial_t}\| \right) + \|\epsilon_t\| \right]. \end{aligned}$$

5 EXPERIMENTAL EVALUATION

In this section we describe experiments aimed at evaluating the utility of EGOP as a metric estimation technique for regression or classification. We consider a family of non-parametric methods that rely on the notion of distance under a given Mahalanobis metric M , computed as $(x - x')^T M (x - x')$. In this setup, we consider three choices of M : (i) identity, i.e., Euclidean distance in the original space; (ii) the estimated gradient weights (GW) matrix as in [6], i.e., Euclidean distance weighted by the estimated $\Delta_{t,i} f_n$, and (iii) the estimated EGOP matrix $\mathbb{E}_n G_n(X)$. The latter corresponds to Euclidean distance in the original space under linear transform given by $[\mathbb{E}_n G_n(X)]^{1/2}$. Note that a major distinction between the metrics based on GW and EGOP is that the former only scales the Euclidean distance, whereas the latter introduces a rotation.

Each choice of M can define the set of neighbors of an input point x in two ways: (a) k nearest neighbors (k NN) of x for a fixed k , or (b) neighbors with distance $\leq h$ for a fixed h ; we will refer to this as h NN. When the task is regression, the output values of the neighbors are simply averaged; for classification, the class label for x is decided by majority vote among neighbors. Note that h NN corresponds to kernel regression with the boxcar kernel.

Thus, we will consider six methods, based on combinations of the choice of metric M and the definition of neighbors: k NN, k NN-GW, k NN-EGOP, h NN, h NN-GW, and h NN-EGOP.

5.1 SYNTHETIC DATA

We first discuss experiments on synthetic data, the goal of which is to examine the effect of varying the dependence of f on input dimensions on the quality of metric recovered with EGOP and alternative approaches. In these experiments, the output is generated i.i.d. as: $y = \sum_i \sin(c_i x_i)$, where the sum is over the dimensions of $x \in \mathbb{R}^d$, and the profile of c determines the degree to which the value of

x_i affects the output. We used $d = 50$ -dimensional input sampled over a bounded domain, and set $c[1] = 50$ and $c[i] = 0.6 * c[i - 1]$ for $i = 2 : 50$. We consider two cases: (R) the input features are transformed by a random rotation in \mathbb{R}^d , *after* y has been generated; and (I) the input features are preserved. Under these conditions we evaluate the out of sample regression accuracy with original metric, GW and EGOP-based metrics, for different value of n ; in each experiment, the values of h and t are tuned by cross-validation on the training set.

The first observation from results in Figures 1 is that adapting the metric by either GW or EGOP helps performance across the board. As can be expected, performance of EGOP, however, is not significantly affected by rotation. On the other hand, GW is able to recover a good metric in the no-rotation case, but much less so under rotation. Some insight into the nature of estimated metrics is obtained from the profile of the estimated feature relevance. For GW this consists of values on the diagonal of \mathbf{M} , and for EGOP of the (square roots) of the eigenvalues of \mathbf{M} . Plots in Figure 1 show these profiles (sorted in descending order). It is clear that EGOP is largely invariant to rotation of the features, and is consistently better at recovering the true relative relevance of features corresponding to the c described above.

5.2 REGRESSION EXPERIMENTS

In this section we present results on several real world datasets. The list of data sets with vital statistics (dimensionality and number of training/test points) is found in Table 1. For each data set, we report the results averaged over ten random training/test splits.

As a measure of performance we compute for each experiment the *normalized mean squared error* (nMSE): mean squared error over test set, divided by target variance over that set. This can be interpreted as fraction of variance in the target unexplained by the regressor.

In each experiment the input was normalized by the mean and standard deviation of the training set. For each method, the values of h or k as well as t (the bandwidth used to estimate finite differences for GW and EGOP) were set by two fold cross-validation on the training set.

5.3 CLASSIFICATION EXPERIMENTS

The setup for classification data sets is very similar for regression, except that the task is binary classification, and the labels of the neighbors selected by each prediction method are aggregated by simple majority vote, rather than averaging as in regression. The performance measure of interest here is classification error. As in regression experiments, we normalized the data, tuned all relevant parameters by cross validation on training data, and repeated the entire experimental procedure ten times with random train-

ing/test splits.

In addition to the baselines listed above, in classification experiments we considered another competitor: the popular feature relevance determination method called ReliefF [4, 5]. A highly engineered method that includes heuristics honed over considerable time by practitioners, it has the same general form of assigning weights to features as do GW and EGOP.

5.4 RESULTS

The detailed results are reported in Tables 1 and 2. These correspond to a single value of training set size. Plots in Figures 2 and 3 show a few representative cases for regression and classification, respectively, of performance of different methods as a function of training set size; it is evident from these that while the performance of all methods tends to improve if additional training data are available, the gaps methods persist across the range of training set sizes.

From the results in Tables 1 and 2, we can see that the -EGOP variants dominate the -GW ones, and that both produce gains relative to using the original metric. This is true both for k NN and for kernel regression (h NN) methods, suggesting general utility of EGOP-based metric, not tied to a particular non-parametric mechanism.

We also see that the metrics based on estimated EGOP are competitive with ReliefF, despite the latter benefiting from extensive engineering efforts over the years.

5.5 EXPERIMENTS WITH LOCAL LINEAR REGRESSION

As mentioned earlier in the paper, our estimator for EGOP is an alternative to an estimator based on computing the slope of locally linear regression (LLR) [1] over the training data. We have compared these two estimation methods on a number of data sets, and the results are plotted in Figure 4. In these experiments, the bandwidth of LLR was tuned by a 2-fold cross-validation on the training data.

We observe that despite its simplicity, the accuracy of predictors using EGOP-based metric estimated by our approach is competitive with or even better than the accuracy with EGOP estimated using LLR. As the sample size increases, accuracy of LLR improves. However, the computational expense of LLR-based estimator also grows with the size of data, and in our experiments it became dramatically slower than our estimator of EGOP for the larger data sizes. This confirms the intuition that our estimator is an appealing alternative to LLR-based estimator, offering a good tradeoff of speed and accuracy.

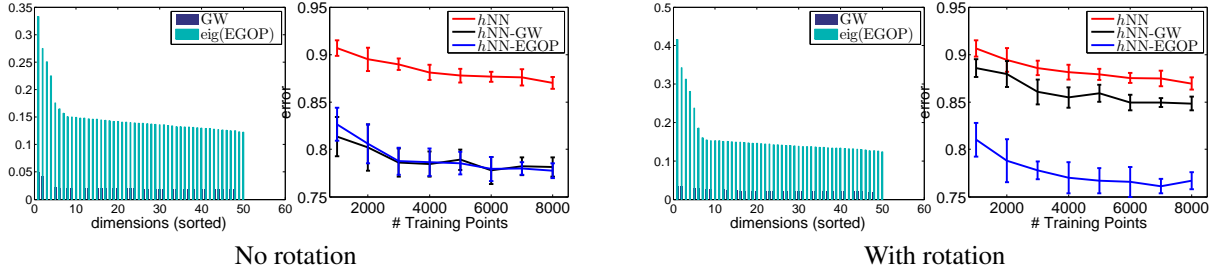


Figure 1: Synthetic data, $d=50$, with and without rotation applied after generating y from x . In each case we show error of hNN with different metrics (left) and the profile of derivatives recovered by GW and EGOP. The deterioration of the error performance of the Gradient Weights approach after the feature space is subject to a random rotation is noteworthy. See text for details.

Table 1: Regression results, with ten random runs per data set.

Dataset	d	train/test	hNN	$hNN-GW$	$hNN-EGOP$
Ailerons	5	3000/2000	0.3637 ± 0.0099	0.3381 ± 0.0087	0.3264 ± 0.0095
Concrete	8	730/300	0.3625 ± 0.0564	0.2525 ± 0.0417	0.2518 ± 0.0418
Housing	13	306/200	0.3033 ± 0.0681	0.2628 ± 0.0652	0.2776 ± 0.0550
Wine	11	2500/2000	0.7107 ± 0.0157	0.7056 ± 0.0184	0.6867 ± 0.0145
Barrett1	21	3000/2000	0.0914 ± 0.0106	0.0740 ± 0.0209	0.0927 ± 0.0322
Barrett5	21	3000/2000	0.0906 ± 0.0044	0.0823 ± 0.0171	0.0996 ± 0.0403
Sarcos1	21	3000/2000	0.1433 ± 0.0087	0.0913 ± 0.0054	0.1064 ± 0.0101
Sarcos5	21	3000/2000	0.1101 ± 0.0033	0.0972 ± 0.0044	0.0970 ± 0.0064
ParkinsonM	19	3000/2000	0.4234 ± 0.0386	0.3606 ± 0.0524	0.3546 ± 0.0406
ParkinsonT	19	3000/2000	0.4965 ± 0.0606	0.3980 ± 0.0738	0.4168 ± 0.0941
TeleComm	48	3000/2000	0.1079 ± 0.0099	0.0858 ± 0.0089	0.0380 ± 0.0059

Dataset	kNN	$kNN-GW$	$kNN-EGOP$
Ailerons	0.3364 ± 0.0087	0.3161 ± 0.0058	0.3154 ± 0.0100
Concrete	0.2884 ± 0.0311	0.2040 ± 0.0234	0.2204 ± 0.0292
Housing	0.2897 ± 0.0632	0.2389 ± 0.0604	0.2546 ± 0.0550
Wine	0.6633 ± 0.0119	0.6615 ± 0.0134	0.6574 ± 0.0171
Barrett1	0.1051 ± 0.0150	0.0843 ± 0.0229	0.1136 ± 0.0510
Barrett5	0.1095 ± 0.0096	0.0984 ± 0.0244	0.1120 ± 0.0315
Sarcos1	0.1222 ± 0.0074	0.0769 ± 0.0037	0.0890 ± 0.0072
Sarcos5	0.0870 ± 0.0051	0.0779 ± 0.0026	0.0752 ± 0.0051
ParkinsonM	0.3638 ± 0.0443	0.3181 ± 0.0477	0.3211 ± 0.0479
ParkinsonT	0.4055 ± 0.0413	0.3587 ± 0.0657	0.3528 ± 0.0742
TeleComm	0.0864 ± 0.0094	0.0688 ± 0.0074	0.0289 ± 0.0031

Table 2: Classification results with 3000 training/2000 testing.

Dataset	d	hNN	$hNN-GW$	$hNN-EGOP$	$hNN-ReliefF$
Cover Type	10	0.2301 ± 0.0104	0.2176 ± 0.0105	0.2197 ± 0.0077	0.1806 ± 0.0165
Gamma	10	0.1784 ± 0.0093	0.1721 ± 0.0082	0.1658 ± 0.0076	0.1696 ± 0.0072
Page Blocks	10	0.0410 ± 0.0042	0.0387 ± 0.0085	0.0383 ± 0.0047	0.0395 ± 0.0053
Shuttle	9	0.0821 ± 0.0095	0.0297 ± 0.0327	0.0123 ± 0.0041	0.1435 ± 0.0458
Musk	166	0.0458 ± 0.0057	0.0477 ± 0.0069	0.0360 ± 0.0037	0.0434 ± 0.0061
IJCNN	22	0.0523 ± 0.0043	0.0452 ± 0.0045	0.0401 ± 0.0039	0.0510 ± 0.0067
RNA	8	0.1128 ± 0.0038	0.0710 ± 0.0048	0.0664 ± 0.0064	0.1343 ± 0.0406

Dataset	kNN	$kNN-GW$	$kNN-EGOP$	$kNN-ReliefF$
Cover Type	0.2279 ± 0.0091	0.2135 ± 0.0064	0.2161 ± 0.0061	0.1839 ± 0.0087
Gamma	0.1775 ± 0.0070	0.1680 ± 0.0075	0.1644 ± 0.0099	0.1623 ± 0.0063
Page Blocks	0.0349 ± 0.0042	0.0361 ± 0.0048	0.0329 ± 0.0033	0.0347 ± 0.0038
Shuttle	0.0037 ± 0.0025	0.0024 ± 0.0016	0.0021 ± 0.0011	0.0028 ± 0.0021
Musk	0.2279 ± 0.0091	0.2135 ± 0.0064	0.2161 ± 0.0061	0.1839 ± 0.0087
IJCNN	0.0540 ± 0.0061	0.0459 ± 0.0058	0.0413 ± 0.0051	0.0535 ± 0.0080
RNA	0.1042 ± 0.0063	0.0673 ± 0.0062	0.0627 ± 0.0057	0.0828 ± 0.0056

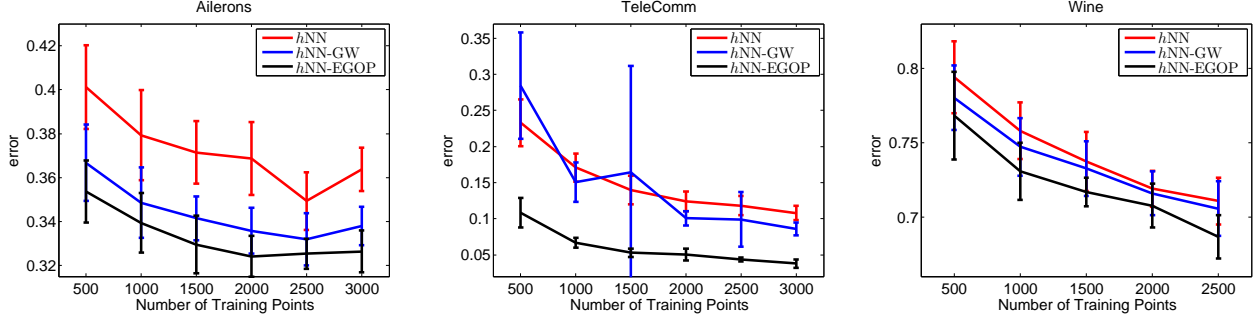


Figure 2: Regression error (nMSE) as a function of training set size for Ailerons, TeleComm, Wine data sets.

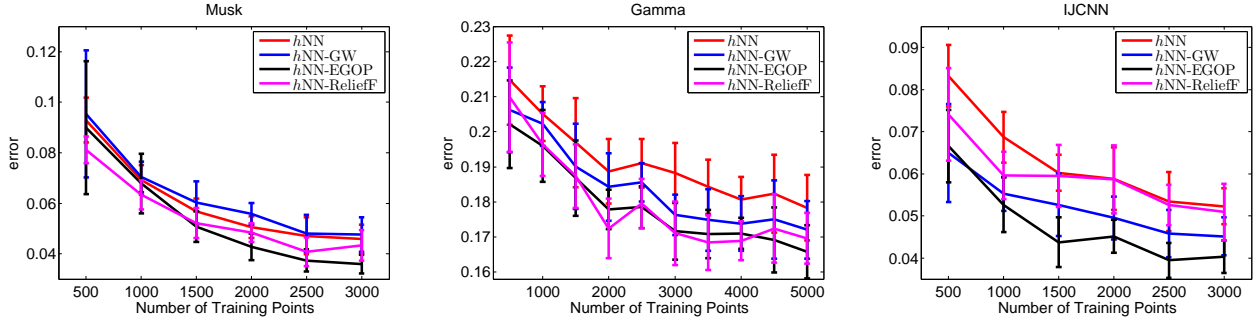


Figure 3: Classification error as a function of training set size for Musk, Gamma, IJCNN data sets.

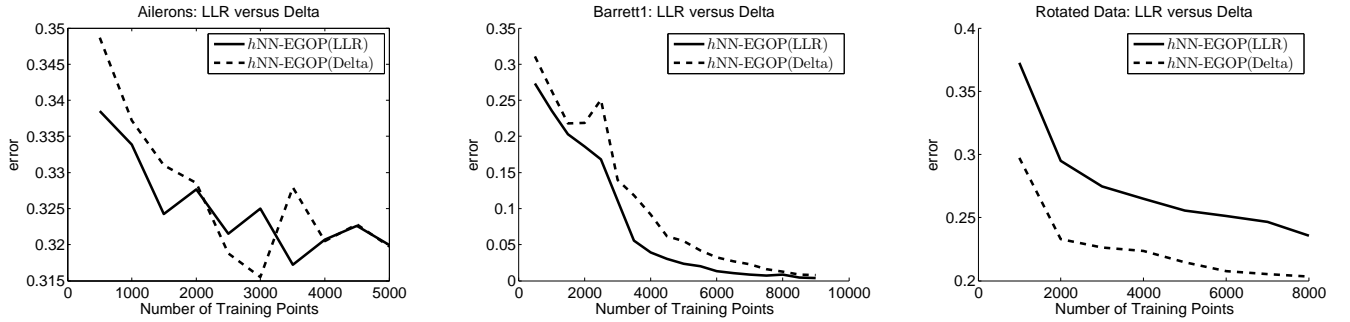


Figure 4: Comparison of EGOP estimated by our proposed method vs. locally linear regression, for Ailerons, Barrett1 and the synthetic (with rotation) data sets (this synthetic dataset is similar to the one used in section 5.1 but with $d = 12$ and $c = [5, 3, 1, .5, .2, .1, .08, .06, .05, .04, .03, .02]$). We also report the following running times (averaged over the ten random runs) for the same using our method and LLR respectively for the highest sample size used in the above real world datasets: Ailerons (128.13s for delta and 347.48s for LLR), Barrett (377.03s for delta and 1650.55s for LLR). Showing that our rough estimator is significantly faster than Local Linear Regression while giving competitive performance. These timings were recorded on an Intel i7 processor with CPU @ 2.40 GHz and 12 GB of RAM.

References

- [1] W. S. Cleveland and S. J. Devlin. Locally weighted regression: an approach to regression analysis by local fitting. *Journal of the American Statistical Association*, 83(403):596–610, 1988.
- [2] W. Hardle, P. Hall, H. Ichimura, et al. Optimal smoothing in single-index models. *The annals of Statistics*, 21(1):157–178, 1993.
- [3] S. Kakade. Lecture notes on multivariate analysis, dimensionality reduction, and spectral methods. *STAT 991*, Spring, 2010.
- [4] K. Kira and L. A. Rendell. The feature selection problem: Traditional methods and a new algorithm. In W. R. Swartout, editor, *AAAI*, pages 129–134. AAAI Press / The MIT Press, 1992.
- [5] I. Kononenko, E. Simec, and M. Robnik-Sikonja. Overcoming the myopia of inductive learning algo-

- rithms with relief. *Applied Intelligence*, 7:39–55, 1997.
- [6] S. Kpotufe and A. Boularias. Gradient weights help nonparametric regressors. In *NIPS*, pages 2870–2878, 2012.
 - [7] K.-C. Li. Sliced inverse regression for dimension reduction. *Journal of the American Statistical Association*, 86(414):316–327, 1991.
 - [8] S. Mukherjee, Q. Wu, D.-X. Zhou, et al. Learning gradients on manifolds. *Bernoulli*, 16(1):181–207, 2010.
 - [9] J. L. Powell, J. H. Stock, and T. M. Stoker. Semi-parametric estimation of index coefficients. *Econometrica: Journal of the Econometric Society*, pages 1403–1430, 1989.
 - [10] J. A. Tropp. User-friendly tools for random matrices: An introduction. *Tutorial at NIPS*, 2012.
 - [11] Q. Wu, J. Guinney, M. Maggioni, and S. Mukherjee. Learning gradients: predictive models that infer geometry and statistical dependence. *The Journal of Machine Learning Research*, 11:2175–2198, 2010.
 - [12] Y. Xia, H. Tong, W. Li, and L.-X. Zhu. An adaptive estimation of dimension reduction space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 64(3):363–410, 2002.

Venn–Abers Predictors

Vladimir Vovk and Ivan Petej
Department of Computer Science
Royal Holloway, University of London
Egham, Surrey, UK

Abstract

This paper continues study, both theoretical and empirical, of the method of Venn prediction, concentrating on binary prediction problems. Venn predictors produce probability-type predictions for the labels of test objects which are guaranteed to be well calibrated under the standard assumption that the observations are generated independently from the same distribution. We give a simple formalization and proof of this property. We also introduce Venn–Abers predictors, a new class of Venn predictors based on the idea of isotonic regression, and report promising empirical results both for Venn–Abers predictors and for their more computationally efficient simplified version.

1 INTRODUCTION

Venn predictors were introduced in [16] and are discussed in detail in [15], Chapter 6, but to make the paper self-contained we define them in Section 2. This section also states the important property of validity of Venn predictors: they are automatically well calibrated. In some form this property of validity has been known: see, e.g., [15], Theorem 6.6. However, this known version is complicated, whereas our version (Theorem 1 below) is much simpler and the intuition behind it is more transparent. In the same section we show (Theorem 2) that Venn prediction is essentially the only way to achieve our new property of validity.

Section 3 defines a natural class of Venn predictors, which we call Venn–Abers predictors (with the “Abers” part formed by the initial letters of the authors’ surnames of the paper [1] introducing the underlying technique). The latter are defined on top of a wide class of classification algorithms, which we call “scoring classifiers” in this paper; each scoring classifier can be automatically transformed into a Venn–Abers predictor, and we refer to this

transformation as the “Venn–Abers method”. Because of its theoretical guarantees, this method can be used for improving the calibration of probabilistic predictions.

The definition of Venn–Abers predictors was prompted by [8], which demonstrated that the method of calibrating probabilistic predictions introduced by Zadrozny and Elkan in [17] (an adaptation of the isotonic regression procedure of [1]) does not always achieve its goal and sometimes leads to poorly calibrated predictions. Another paper reporting the possibility for the Zadrozny–Elkan method to produce grossly miscalibrated predictions is [7]. The Venn–Abers method is a simple modification of Zadrozny and Elkan’s method; being a special case of Venn prediction, it overcomes the problem of potentially poor calibration.

Theorem 1 in Section 2 says that Venn predictors are perfectly calibrated. The price to pay, however, is that Venn predictors are multiprobabilistic predictors, in the sense of issuing a set of probabilistic predictions instead of a single probabilistic prediction; intuitively, the diameter of this set reflects the uncertainty of our prediction. In Section 5 we explore the efficiency of Venn–Abers predictors empirically using the fundamental log loss function and another popular loss function, square loss. To apply these loss functions, we need, however, probabilistic predictions rather than multiprobabilistic predictions, and in Section 4 we define natural minimax ways of replacing the latter with the former.

In Section 5 we explore the empirical predictive performance of the most natural version of the original Zadrozny–Elkan method, the Venn–Abers method, and the latter’s simplified version, which is not only simpler but also more efficient computationally. We use nine benchmark data sets from the UCI repository [5] and six standard scoring classifiers, and for each combination of a data set and classifier evaluate the predictive performance of each method. Our results show that the Venn–Abers and simplified Venn–Abers methods usually improve the performance of the underlying classifiers, and in our experiments they work better than the original Zadrozny–Elkan

method.

Interestingly, the predictive performance of the simplified Venn–Abers method is slightly better than that of the Venn–Abers method on the chosen data sets and scoring classifiers; for example, in the case of the log loss function, the best performance is achieved by the simplified Venn–Abers methods for seven data sets out of the nine. If these results are confirmed in wider empirical studies, the simplified Venn–Abers method is preferred since it achieves both computational and predictive efficiency.

Our empirical study in Section 5 does not mean that we recommend that the multiprobabilistic predictions output by Venn–Abers (and more generally Venn) predictors be replaced by probabilistic predictions (e.g., using the formulas of Section 4). On the contrary, we believe that the size of a multiprobabilistic prediction carries valuable information about the uncertainty of the prediction. The only purpose of replacing multiprobabilistic by probabilistic predictions is to facilitate comparison of various prediction algorithms using well-established loss functions.

2 VENN PREDICTORS

We consider *observations* $z = (x, y)$ consisting of two components: an *object* $x \in \mathbf{X}$ and its *label* $y \in \mathbf{Y}$. In this paper we are only interested in the binary case and for concreteness set $\mathbf{Y} := \{0, 1\}$. We assume that \mathbf{X} is a measurable space, so that observations are elements of the measurable space that is the Cartesian product $\mathbf{Z} := \mathbf{X} \times \mathbf{Y} = \mathbf{X} \times \{0, 1\}$.

A *Venn taxonomy* A is a measurable function that assigns to each $n \in \{2, 3, \dots\}$ and each sequence $(z_1, \dots, z_n) \in \mathbf{Z}^n$ an equivalence relation \sim on $\{1, \dots, n\}$ which is equivariant in the sense that, for each n and each permutation π of $\{1, \dots, n\}$,

$$(i \sim j \mid z_1, \dots, z_n) \implies (\pi(i) \sim \pi(j) \mid z_{\pi(1)}, \dots, z_{\pi(n)}),$$

where the notation $(i \sim j \mid z_1, \dots, z_n)$ means that i is equivalent to j under the relation assigned by A to (z_1, \dots, z_n) . The measurability of A means that for all n, i , and j the set $\{(z_1, \dots, z_n) \mid (i \sim j \mid z_1, \dots, z_n)\}$ is measurable. Define

$$\begin{aligned} A(j \mid z_1, \dots, z_n) \\ := \{i \in \{1, \dots, n\} \mid (i \sim j \mid z_1, \dots, z_n)\} \end{aligned}$$

to be the equivalence class of j . Let (z_1, \dots, z_l) be a training sequence of observations $z_i = (x_i, y_i)$, $i = 1, \dots, l$, and x be a test object. The *Venn predictor* associated with a given Venn taxonomy A outputs the pair (p_0, p_1) as its prediction for x 's label, where

$$p_y := \frac{|\{i \in A(l+1 \mid z_1, \dots, z_l, (x, y)) \mid y_i = 1\}|}{|A(l+1 \mid z_1, \dots, z_l, (x, y))|}$$

for both $y \in \{0, 1\}$ (notice that the denominator is always positive). Intuitively, p_0 and p_1 are the predicted probabilities that the label of x is 1; of course, the prediction is useful only when $p_0 \approx p_1$. The *probability interval* output by a Venn predictor is defined to be the convex hull $\text{conv}(p_0, p_1)$ of the set $\{p_0, p_1\}$; we will sometimes refer to the pair (p_0, p_1) or the set $\{p_0, p_1\}$ as the *multiprobabilistic prediction*.

Validity of Venn predictors

Let us say that a random variable P taking values in $[0, 1]$ is *perfectly calibrated* for a random variable Y taking values in $\{0, 1\}$ if

$$\mathbb{E}(Y \mid P) = P \quad \text{a.s.} \quad (1)$$

Intuitively, P is the prediction made by a probabilistic predictor for Y , and perfect calibration means that the probabilistic predictor gets the probabilities right, at least on average, for each value of the prediction. A probabilistic predictor for Y whose prediction P satisfies (1) with an approximate equality is said to be well calibrated [4], or unbiased in the small [11, 4]; this terminology will be used only in informal discussions, of course.

A *selector* is a random variable taking values 0 or 1.

Theorem 1. *Let $(X_1, Y_1), \dots, (X_l, Y_l), (X, Y)$ be IID (independent identically distributed) random observations. Fix a Venn predictor V and an $l \in \{1, 2, \dots\}$. Let (P_0, P_1) be the output of V given $(X_1, Y_1, \dots, X_l, Y_l)$ as the training set and X as the test object. There exists a selector S such that P_S is perfectly calibrated for Y .*

Intuitively, at least one of the two probabilities output by the Venn predictor is perfectly calibrated. Therefore, if the two probabilities tend to be close to each other, we expect them (or, say, their average) to be well calibrated.

In the proof of Theorem 1 and later in the paper we will use the notation $\{a_1, \dots, a_n\}$ for bags (in other words, multisets); the cardinality of the set $\{a_1, \dots, a_n\}$ might well be smaller than n (because of the removal of all duplicates in the bag). Intuitively, $\{a_1, \dots, a_n\}$ is the sequence (a_1, \dots, a_n) with its ordering forgotten. We will sometimes refer to the bag $\{z_1, \dots, z_l\}$, where (z_1, \dots, z_l) is the training sequence, as the training set (although technically it is a multiset rather than a set).

Proof of Theorem 1. Take $S := Y$ as the selector. Let us check that (1) is true even if we further condition on the observed bag $\{ (X_1, Y_1), \dots, (X_l, Y_l), (X, Y) \}$ (so that the remaining randomness consists in generating a random permutation of this bag). We only need to check the equality $\mathbb{E}(Y \mid P = p) = p$, where P is the average of 1s in the equivalence class containing (X, Y) , for the p s which are the percentages of 1s in various equivalence classes (further

conditioning on the observed bag is not reflected in our notation). For each such p , $\mathbb{E}(Y \mid P = p)$ is the average of 1s in the equivalence classes for which the average of 1s is p ; therefore, we indeed have $\mathbb{E}(Y \mid P = p) = p$. \square

The following simple corollary of Theorem 1 gives a weaker property of validity, which is sometimes called “unbiasedness in the large” [11, 4].

Corollary 1. *For any Venn predictor V and any $l = 1, 2, \dots$,*

$$\mathbb{P}(Y = 1) \in [\mathbb{E}(\underline{V}(X; X_1, Y_1, \dots, X_l, Y_l)), \mathbb{E}(\overline{V}(X; X_1, Y_1, \dots, X_l, Y_l))], \quad (2)$$

where $(X_1, Y_1), \dots, (X_l, Y_l), (X, Y)$ are IID observations and $[\underline{V}(\dots), \overline{V}(\dots)]$ is the probability interval produced by V for the test object X based on the training sequence $(X_1, Y_1, \dots, X_l, Y_l)$.

Proof. It suffices to notice that, for a selector S such that $P = P_S$ ((P_0, P_1) being the output of V) satisfies the condition of perfect calibration (1),

$$\begin{aligned} \mathbb{P}(Y = 1) &= \mathbb{E}(Y) = \mathbb{E}(\mathbb{E}(Y \mid P_S)) \\ &= \mathbb{E}(P_S) \in [\mathbb{E}\underline{V}, \mathbb{E}\overline{V}], \end{aligned}$$

where the arguments of \underline{V} and \overline{V} are omitted. \square

Unbiasedness in the large (2) is easy to achieve even for probabilistic predictors if we do not care about other measures of quality of our predictions: for example, the probabilistic predictor ignoring the x s and outputting k/l as its prediction, where k is the number of 1s in the training sequence of size l , is unbiased in the large. Unbiasedness in the small (1) is also easy to achieve if we allow multiprobabilistic predictors: consider the multiprobabilistic predictor ignoring the x s and outputting $\{k/(l+1), (k+1)/(l+1)\}$ as its prediction. The problem is how to achieve predictive efficiency (making our prediction as relevant to the test object as possible without overfitting) while maintaining validity.

Our following result, Theorem 2, will say that under mild regularity conditions unbiasedness in the small (1) holds only for Venn predictors (perhaps weakened by adding irrelevant probabilistic predictions) and, therefore, implies all other properties of validity, such as the more complicated one given in [15, Chapter 6].

To state Theorem 2 we need a few further definitions. Let us fix the length l of the training sequence for now. A *multiprobabilistic predictor* is a function that maps each sequence $(z_1, \dots, z_l) \in \mathbf{Z}^l$ to a subset of $[0, 1]$ (not required to be measurable in any sense). Venn predictors are an important example for this paper. Let us say that a multiprobabilistic predictor is *invariant* if it is independent of the ordering of the training set (z_1, \dots, z_l) . An *invariant selector*

for an invariant multiprobabilistic predictor F is a measurable function $f : \mathbf{Z}^{l+1} \rightarrow [0, 1]$ such that $f(z_1, \dots, z_{l+1})$ does not change when z_1, \dots, z_l are permuted and such that $f(z_1, \dots, z_{l+1}) \in F(z_1, \dots, z_l)$ for all (z_1, \dots, z_{l+1}) . (It is natural to consider only invariant predictors and selectors under the IID assumption because of the principle of sufficiency [3, Chap. 2].) We say that an invariant multiprobabilistic predictor F is *invariantly perfectly calibrated* if it has an invariant selector f such that

$$\begin{aligned} \mathbb{E}(Y \mid f(Z_1, \dots, Z_l, (X, Y))) \\ = f(Z_1, \dots, Z_l, (X, Y)) \quad \text{a.s.} \end{aligned} \quad (3)$$

whenever $Z_1, \dots, Z_l, (X, Y)$ are IID observations.

Theorem 2. *If an invariant multiprobabilistic predictor F is invariantly perfectly calibrated, then it contains a Venn predictor V in the sense that both elements of $V(Z_1, \dots, Z_l)$ belong to $F(Z_1, \dots, Z_l)$ almost surely provided Z_1, \dots, Z_l are IID.*

Proof. Let f be an invariant selector of F satisfying the condition (3) of being invariantly perfectly calibrated. By definition,

$$\begin{aligned} \mathbb{E}(Y - f(Z_1, \dots, Z_l, (X, Y)) \mid \\ f(Z_1, \dots, Z_l, (X, Y))) = 0 \quad \text{a.s.}, \end{aligned}$$

which implies

$$\begin{aligned} \mathbb{E}((Y - f(Z_1, \dots, Z_l, (X, Y))) \\ 1_{\{f(Z_1, \dots, Z_l, (X, Y)) \in [a, b]\}}) = 0 \quad \text{a.s.} \end{aligned} \quad (4)$$

for all intervals $[a, b]$ with rational end-points. The expected value in (4) can be obtained in two steps: first we average

$$(y'_{l+1} - f(z'_1, \dots, z'_{l+1})) 1_{\{f(z'_1, \dots, z'_{l+1}) \in [a, b]\}}$$

over the orderings (z'_1, \dots, z'_{l+1}) of each bag $\{z_1, \dots, z_{l+1}\}$, where $z_i = (x_i, y_i)$ and $z'_i = (x'_i, y'_i)$, and then we average over the bags $\{z_1, \dots, z_{l+1}\}$ generated according to $z_i := Z_i$, $i = 1, \dots, l$, and $z_{l+1} := (X, Y)$. The first operation is discrete: the average over the orderings of $\{z_1, \dots, z_{l+1}\}$ is the arithmetic mean of $(y_i - p_i) 1_{\{p_i \in [a, b]\}}$ over $i = 1, \dots, l+1$, where $p_i := f(\dots, z_i)$ and the dots stand for z_1, \dots, z_{i-1} and z_{i+1}, \dots, z_{l+1} arranged in any order (since f is invariant, the order does not matter). By the completeness of the statistic that maps a data sequence of size $l+1$ to the corresponding bag [10, Section 4.3], this average is zero for all $[a, b]$ and almost all bags. Without loss of generality we assume that this holds for all bags.

Define a Venn taxonomy A as follows: given a sequence (z_1, \dots, z_{l+1}) , set $i \sim j$ if $p_i = p_j$ where p is defined as above. It is easy to check that the corresponding Venn predictor satisfies the requirement in Theorem 2. \square

Remark. The invariance assumption in Theorem 2 is essential. Indeed, suppose $l > 1$ and consider the multiprobabilistic predictor whose prediction for the label of the test observation does not depend on the objects and is

$$\begin{cases} \{k/l, (k+1)/l\} & \text{if } y_1 = 0 \\ \{(k-1)/l, k/l\} & \text{if } y_1 = 1, \end{cases}$$

where k is the number of 1s among the labels of the l training observations. This non-invariant predictor is perfectly calibrated (see below) but does not contain a Venn predictor (if it did, such a Venn predictor, being invariant, would always output the one-element multiprobabilistic prediction $\{k/l\}$, which is impossible). Let us check that this non-invariant predictor is indeed perfectly calibrated, even given the union of the training set and the test observation (i.e., given the bag of size $l+1$ obtained from the training sequence by joining the test observation and then forgetting the ordering). Take the selector such that the selected probabilistic predictor is

$$\begin{cases} k/l & \text{for sequences of the form } 0 \dots 0 \\ (k+1)/l & \text{for sequences of the form } 0 \dots 1 \\ (k-1)/l & \text{for sequences of the form } 1 \dots 0 \\ k/l & \text{for sequences of the form } 1 \dots 1. \end{cases}$$

For a binary sequence of labels of length $l+1$ with m 1s the probabilistic prediction P for its last element will be, therefore,

$$\begin{cases} m/l & \text{for sequences of the form } 0 \dots 0 \\ m/l & \text{for sequences of the form } 0 \dots 1 \\ (m-1)/l & \text{for sequences of the form } 1 \dots 0 \\ (m-1)/l & \text{for sequences of the form } 1 \dots 1. \end{cases}$$

The conditional probability that $Y = 1$ (Y being the label of the last element) given $P = p$ (and given m) is

$$\frac{\binom{l-1}{m-1}}{\binom{l}{m}} = \frac{m}{l}$$

when $p = m/l$ and is

$$\frac{\binom{l-1}{m-2}}{\binom{l}{m-1}} = \frac{m-1}{l}$$

when $p = (m-1)/l$; in both cases we have perfect calibration.

3 VENN-ABERS PREDICTORS

We say that a function f is *increasing* if its domain is an ordered set and $t_1 \leq t_2 \Rightarrow f(t_1) \leq f(t_2)$.

Many machine-learning algorithms for classification are in fact *scoring classifiers*: when trained on a training sequence of observations and fed with a test object x , they

output a *prediction score* $s(x)$; we will call $s : \mathbf{X} \rightarrow \mathbb{R}$ the *scoring function* for that training sequence. The actual classification algorithm is obtained by fixing a threshold c and predicting the label of x to be 1 if and only if $s(x) \geq c$ (or if and only if $s(x) > c$). Alternatively, one could apply an increasing function g to $s(x)$ in an attempt to “calibrate” the scores, so that $g(s(x))$ can be used as the predicted probability that the label of x is 1.

Fix a scoring classifier and let (z_1, \dots, z_l) be a training sequence of observations $z_i = (x_i, y_i)$, $i = 1, \dots, l$. The most direct application [17] of the method of isotonic regression [1] to the problem of score calibration is as follows. Train the scoring classifier on the training sequence and compute the score $s(x_i)$ for each training observation (x_i, y_i) , where s is the scoring function for (z_1, \dots, z_l) . Let g be the increasing function on the set $\{s(x_1), \dots, s(x_l)\}$ that maximizes the likelihood

$$\prod_{i=1}^l p_i, \quad \text{where } p_i := \begin{cases} g(s(x_i)) & \text{if } y_i = 1 \\ 1 - g(s(x_i)) & \text{if } y_i = 0. \end{cases} \quad (5)$$

Such a function g is indeed unique [1, Corollary 2.1] and can be easily found using the “pair-adjacent violators algorithm” (PAVA, described in detail in the summary of [1] and in [2, Section 1.2]; see also the proof of Lemma 1 below). We will say that g is the *isotonic calibrator* for $((s(x_1), y_1), \dots, (s(x_l), y_l))$. To predict the label of a test object x , the direct method finds the closest $s(x_i)$ to $s(x)$ and outputs $g(s(x_i))$ as its prediction (in the case of ties our implementation of this method used in Section 5 chooses the smaller $s(x_i)$; however, ties almost never happen in our experiments). We will refer to this as the *direct isotonic regression* (DIR) method.

The direct method is prone to overfitting as the same observations z_1, \dots, z_l are used both for training the scoring classifier and for calibration without taking any precautions. The *Venn-Abers predictor* corresponding to the given scoring classifier is the multiprobabilistic predictor that is defined as follows. Try the two different labels, 0 and 1, for the test object x . Let s_0 be the scoring function for $(z_1, \dots, z_l, (x, 0))$, s_1 be the scoring function for $(z_1, \dots, z_l, (x, 1))$, g_0 be the isotonic calibrator for

$$((s_0(x_1), y_1), \dots, (s_0(x_l), y_l), (s_0(x), 0)), \quad (6)$$

and g_1 be the isotonic calibrator for

$$((s_1(x_1), y_1), \dots, (s_1(x_l), y_l), (s_1(x), 1)). \quad (7)$$

The multiprobabilistic prediction output by the Venn-Abers predictor is (p_0, p_1) , where $p_0 := g_0(s_0(x))$ and $p_1 := g_1(s_1(x))$. (And we can expect p_0 and p_1 to be close to each other unless DIR overfits grossly.) The Venn-Abers predictor is described as Algorithm 1.

The intuition behind Algorithm 1 is that it tries to evaluate the robustness of the DIR prediction. To see how sensitive the scoring function is to the training set we extend

Algorithm 1 Venn–Abers predictor

Input: training sequence (z_1, \dots, z_l) **Input:** test object x **Output:** multiprobabilistic prediction (p_0, p_1) **for** $y \in \{0, 1\}$ **do** set s_y to the scoring function for $(z_1, \dots, z_l, (x, y))$ set g_y to the isotonic calibrator for $(s_y(x_1), y_1), \dots, (s_y(x_l), y_l), (s_y(x), y))$ set $p_y := g_y(s_y(x))$ **end for**

the latter by adding the test object labelled in two different ways. And to see how sensitive the probabilistic prediction is, we again consider the training set extended in two different ways (if it is sensitive, the prediction will be fragile even if the scoring function is robust). For large data sets and inflexible scoring functions, we will have $p_0 \approx p_1$, and both numbers will be close to the DIR prediction. However, even if the data set is very large but the scoring function is very flexible, p_0 can be far from p_1 (the extreme case is where the scoring function is so flexible that it ignores all observations apart from a few that are most similar to the test object, and in this case it does not matter how big the data set is). We rarely know in advance how flexible our scoring function is relative to the size of the data set, and the difference between p_0 and p_1 gives us some indication of this.

The following proposition says that Venn–Abers predictors are Venn predictors and, therefore, inherit all properties of validity of the latter, such as Theorem 1.

Proposition 1. *Venn–Abers predictors are Venn predictors.*

Proof. Fix a Venn–Abers predictor. The corresponding Venn taxonomy is defined as follows: given a sequence

$$(z_1, \dots, z_n) = ((x_1, y_1), \dots, (x_n, y_n)) \in (\mathbf{X} \times \{0, 1\})^n$$

and $i, j \in \{1, \dots, n\}$, we set $i \sim j$ if and only if $g(s(x_i)) = g(s(x_j))$, where s is the scoring function for (z_1, \dots, z_n) and g is the isotonic calibrator for

$$((s(x_1), y_1), \dots, (s(x_n), y_n)).$$

Lemma 1 below shows that the Venn predictor corresponding to this taxonomy gives predictions identical to those given by the original Venn–Abers predictor. This proves the proposition. \square

Lemma 1. *Let g be the isotonic calibrator for*

$$((t_1, y_1), \dots, (t_n, y_n)),$$

where $t_i \in \mathbb{R}$ and $y_i \in \{0, 1\}$, $i = 1, \dots, n$. Any $p \in \{g(t_1), \dots, g(t_n)\}$ is equal to the arithmetic mean of the labels y_i of the t_i , $i = 1, \dots, n$, satisfying $g(t_i) = p$.

Proof. The statement of the lemma immediately follows from the definition of the PAVA [1, summary], which we will reproduce here. Arrange the numbers t_i in the strictly increasing order $t_{(1)} < \dots < t_{(k)}$, where $k \leq n$ is the number of distinct elements among t_i . We would like to find the increasing function g on the set $\{t_{(1)}, \dots, t_{(k)}\} = \{t_1, \dots, t_n\}$ maximizing the likelihood (defined by (5) with t_i in place of $s(x_i)$ and n in place of l). The procedure is recursive. At each step the set $\{t_{(1)}, \dots, t_{(k)}\}$ is partitioned into a number of disjoint cells consisting of adjacent elements of the set; to each cell is assigned a ratio a/N (formally, a pair of integers, with $a \geq 0$ and $N > 0$); the function g defined at this step (perhaps to be redefined at the following steps) is constant on each cell. For $j = 1, \dots, k$, let $a_{(j)}$ be the number of i such that $y_i = 1$ and $t_i = t_{(j)}$, and let $N_{(j)}$ be the number of i such that $t_i = t_{(j)}$. Start from the partition of $\{t_{(1)}, \dots, t_{(k)}\}$ into one-element cells, assign the ratio $a_{(j)}/N_{(j)}$ to $\{t_{(j)}\}$, and set

$$g(t_{(j)}) := \frac{a_{(j)}}{N_{(j)}} \quad (8)$$

(in the notation used in this proof, a/N is a pair of integers whereas $\frac{a}{N}$ is a rational number, the result of the division). If the function g is increasing, we are done. If not, there is a pair C_1, C_2 of adjacent cells (“violators”) such that C_1 is to the left of C_2 and $g(C_1) > g(C_2)$ (where $g(C)$ stands for the common value of $g(t_{(j)})$ for $t_{(j)} \in C$); in this case redefine the partition by merging C_1 and C_2 into one cell C , assigning the ratio $(a_1 + a_2)/(N_1 + N_2)$ to C , where a_1/N_1 and a_2/N_2 are the ratios assigned to C_1 and C_2 , respectively, and setting

$$\begin{aligned} g(t_{(j)}) &:= \frac{N_1}{N_1 + N_2} g(C_1) + \frac{N_2}{N_1 + N_2} g(C_2) \\ &= \frac{a_1 + a_2}{N_1 + N_2} \end{aligned} \quad (9)$$

for all $t_{(j)} \in C$. Repeat the process until g becomes increasing (the number of cells decreases by 1 at each iteration, so the process will terminate in at most k steps). The final function g is the one that maximizes the likelihood. The statement of the lemma follows from this recursive definition: it is true by definition for the initial function (8) and remains true when g is redefined by (9). \square

4 PROBABILISTIC PREDICTORS DERIVED FROM VENN PREDICTORS

In the next section we will compare Venn–Abers predictors with known probabilistic predictors using standard loss functions. Since Venn–Abers predictors output pairs of probabilities rather than point probabilities, we will need to fit them (somewhat artificially) in the standard framework extracting one probability p from p_0 and p_1 .

In this paper we will use two loss functions, log loss and square loss. The *log loss* suffered when predicting $p \in [0, 1]$ whereas the true label is y is

$$\lambda_{\ln}(p, y) := \begin{cases} -\ln(1-p) & \text{if } y = 0 \\ -\ln p & \text{if } y = 1. \end{cases}$$

This is the most fundamental loss function, since the cumulative loss $\sum_{i=1}^n \lambda_{\ln}(p_i, y_i)$ over a test sequence of size n is equal to the minus log of the probability that the predictor assigns to the sequence (this assumes either the batch mode of prediction with independent test observations or the on-line mode of prediction); therefore, a smaller cumulative log loss corresponds to a larger probability. The *square loss* suffered when predicting $p \in [0, 1]$ for the true label y is

$$\lambda_{\text{sq}}(p, y) := (y - p)^2.$$

The main advantage of this loss function is that it is *proper* (see, e.g., [4]): the function $\mathbb{E}_{y \sim B_p} \lambda_{\text{sq}}(q, y)$ of $q \in [0, 1]$, where B_p is the Bernoulli distribution with parameter p , attains its minimum at $q = p$. (Of course, the log loss function is also proper.)

First suppose that our loss function is λ_{\ln} and we are given a multiprobabilistic prediction (p_0, p_1) ; let us find the corresponding minimax probabilistic prediction p . If the true outcome is $y = 0$, our regret for using p instead of the appropriate p_0 is $-\ln(1-p) + \ln(1-p_0)$. If $y = 1$, our regret for using p instead of the appropriate p_1 is $-\ln p + \ln p_1$. The first regret as a function of $p \in [0, 1]$ strictly increases from a nonpositive value to ∞ as p changes from 0 to 1. The second regret as a function of p strictly decreases from ∞ to a nonpositive value as p changes from 0 to 1. Therefore, the minimax regret is the solution to

$$-\ln(1-p) + \ln(1-p_0) = -\ln p + \ln p_1,$$

which is

$$p = \frac{p_1}{1 - p_0 + p_1}. \quad (10)$$

The intuition behind this minimax value of p is that we can interpret the multiprobabilistic prediction (p_0, p_1) as the unnormalized probability distribution P on $\{0, 1\}$ such that $P(\{0\}) = 1 - p_0$ and $P(\{1\}) = p_1$; we then normalize P to get a genuine probability distribution $P' := P/P(\{0, 1\})$, and the p in (10) is equal to $P'(\{1\})$. Of course, it is always true that $p \in \text{conv}(p_0, p_1)$.

In the case of the square loss function, the regret is

$$\begin{cases} p^2 - p_0^2 & \text{if } y = 0 \\ (1-p)^2 - (1-p_1)^2 & \text{if } y = 1 \end{cases}$$

and the two regrets are equal when

$$p := p_1 + p_0^2/2 - p_1^2/2. \quad (11)$$

To see how natural this expression is notice that (11) is equivalent to

$$p = \bar{p} + (p_1 - p_0) \left(\frac{1}{2} - \bar{p} \right),$$

where $\bar{p} := (p_0 + p_1)/2$. Therefore, p is a regularized version of \bar{p} : we move \bar{p} towards the neutral value $1/2$ in the typical (for the Venn–Abers method) case where $p_0 < p_1$. In any case, we always have $p \in \text{conv}(p_0, p_1)$.

The following lemma shows that log loss is never infinite for probabilistic predictors derived from Venn predictors.

Lemma 2. *Neither of the methods discussed in this section (see (10) and (11)) ever produces $p \in \{0, 1\}$ when applied to Venn–Abers predictors.*

Proof. Lemma 1 implies that $p_0 < 1$ and that $p_1 > 0$. It remains to notice that both (10) and (11) produce p in the interior of $\text{conv}(p_0, p_1)$ if $p_0 \neq p_1$ and produce $p = p_0 = p_1$ if $p_0 = p_1$ (and this is true for any sensible averaging method). \square

5 EXPERIMENTAL RESULTS

In this section we compare various calibration methods discussed so far by applying them to six standard scoring classifiers (we will usually omit “scoring” in this section) available within Weka [6], a machine learning tool developed at the University of Waikato, NZ. The standard classifiers are J48 decision trees (abbreviated to J48, or even to J), J48 decision trees with bagging (J48 Bagging, or JB), logistic regression (LR), naïve Bayes (NB), neural networks (NN), and support vector machines calibrated using a sigmoid function as defined by Platt [13] (SVM Platt, or simply SVM). Each of these standard classifiers produces scores in the interval $[0, 1]$, which can then be used as probabilistic predictions; however, in most previous studies these have been found to be inaccurate (see [17] and [9]). We use the scores generated by classifiers as inputs, and by applying the DIR (defined in Section 3), Venn–Abers (VA), and simplified Venn–Abers (SVA, see below) methods we investigate how well we can calibrate the scores and improve them in their role as probabilistic predictions.

In the set of experiments described in this section we do not perform a direct comparison to the method developed by Langford and Zadrozny [9] primarily because, as far as we are aware, the algorithms described in their work are not publicly available.

For the purpose of comparison we use a total of nine datasets with binary labels (encoded as 0 or 1) obtained from the UCI machine learning repository [5]: Australian Credit (which we abbreviate to Australian), Breast Cancer (Breast), Diabetes, Echocardiogram (Echo), Hepatitis, Ionosphere, Labor Relations (Labor), Liver Disorders

Table 1: Log loss (MLE) results obtained using standard Weka classifiers (W) and the three calibration methods (VA, SVA, DIR) applied to the standard classifiers’ outputs for the following Weka classifiers: J48, J48 Bagging, logistic regression (upper part) and naïve Bayes, neural networks, and SVM Platt (lower part). The best results for each pair (classifier, dataset) are in bold.

	J48 (J)				J48 Bagging (JB)				logistic regression (LR)			
	W	VA	SVA	DIR	W	VA	SVA	DIR	W	VA	SVA	DIR
Australian	∞	0.380	0.469	∞	0.328	0.369	0.344	∞	0.342	0.340	0.340	∞
Breast	∞	0.607	0.642	∞	0.581	0.592	0.636	∞	0.584	0.567	0.586	∞
Diabetes	∞	0.552	0.635	∞	0.504	0.515	0.561	∞	0.492	0.490	0.491	∞
Echo	∞	0.606	0.670	∞	0.556	0.517	0.563	∞	∞	0.589	0.606	∞
Hepatitis	∞	0.491	0.528	∞	0.420	0.456	0.434	∞	∞	0.393	0.504	∞
Ionosphere	∞	0.383	0.410	∞	∞	0.387	0.251	∞	∞	0.387	0.524	∞
Labor	∞	0.503	0.537	∞	0.427	0.427	0.385	∞	1.927	0.687	0.297	∞
Liver	∞	0.662	0.866	∞	0.609	0.635	0.707	∞	0.619	0.622	0.611	∞
Vote	∞	0.134	0.145	∞	0.135	0.159	0.131	∞	1.059	0.188	0.148	∞

	naïve Bayes (NB)				neural networks (NN)				SVM Platt (SVM)			
	W	VA	SVA	DIR	W	VA	SVA	DIR	W	VA	SVA	DIR
Australian	0.839	0.355	0.367	∞	0.557	0.427	0.450	∞	0.391	0.356	0.351	∞
Breast	0.663	0.563	0.551	∞	0.774	0.615	0.738	∞	0.583	0.568	0.582	∞
Diabetes	0.753	0.495	0.508	∞	0.536	0.500	0.519	∞	0.491	0.497	0.490	∞
Echo	0.658	0.505	0.522	∞	0.770	0.578	0.605	∞	0.558	0.495	0.538	∞
Hepatitis	0.936	0.365	0.372	∞	0.753	0.471	0.484	∞	0.435	0.349	0.404	∞
Ionosphere	0.704	0.262	0.227	∞	0.625	0.427	0.379	∞	0.359	0.250	0.333	∞
Labor	1.854	0.410	0.296	∞	0.325	0.560	0.298	∞	3.643	0.364	0.287	∞
Liver	0.727	0.649	0.661	∞	0.642	0.603	0.615	∞	0.645	0.663	0.639	∞
Vote	0.594	0.218	0.211	∞	0.235	0.229	0.158	∞	0.125	0.211	0.121	∞

(Liver), and Congressional Voting (Vote). The datasets vary in size as well as the number and type of attributes in order to give a reasonable range of conditions encountered in practice.

In our comparison we use the two standard loss functions discussed in the previous section. Namely, on a given test sequence of length n we will calculate the *mean log error* (MLE)

$$\frac{1}{n} \sum_{i=1}^n \lambda_{\ln}(p_i, y_i) \quad (12)$$

and the *root mean square error* (RMSE)

$$\sqrt{\frac{1}{n} \sum_{i=1}^n \lambda_{\text{sq}}(p_i, y_i)}, \quad (13)$$

where p_i is the probabilistic prediction for the label y_i of the i th observation in the test sequence. MLE (12) can be infinite, namely when predicting $p_i \in \{0, 1\}$ while being incorrect. It therefore penalises the overly confident probabilistic predictions much more significantly than RMSE. We compare the performance of the standard classifiers with their versions calibrated using the three methods (VA, SVA, and DIR) under both loss functions for each dataset. In each experiment we randomly permute the dataset and use the first 2/3 observations for training and the remaining 1/3 for testing.

One of the potential drawbacks of the Venn–Abers method is its computational inefficiency: for each test object the

scores have to be recalculated for the training sequence extended by including the test object first labelled as 0 and then labelled as 1. This implies that the total calculation time is at least $2n$ times that of the underlying classifier, where n is the number of test observations. Therefore, we define a simplified version of Venn–Abers predictors, for which the scores are calculated only once without recalculating them for each test object with postulated labels 0 and 1.

In detail, the *simplified Venn–Abers predictor* for a given scoring classifier is defined as follows. Let (z_1, \dots, z_l) be a training sequence and x be a test object. Define s to be the scoring function for (z_1, \dots, z_l) , g_0 to be the isotonic calibrator for

$$((s(x_1), y_1), \dots, (s(x_l), y_l), (s(x), 0)),$$

and g_1 to be the isotonic calibrator for

$$((s(x_1), y_1), \dots, (s(x_l), y_l), (s(x), 1)))$$

(cf. (6) and (7)). The multiprobabilistic prediction output for the label of x by the simplified Venn–Abers (SVA) predictor is (p_0, p_1) , where $p_0 := g_0(s(x))$ and $p_1 := g_1(s(x))$. This method, summarized as Algorithm 2, is intermediate between DIR and the Venn–Abers method.

Notice that Lemma 2 continues to hold for SVA predictors; therefore, they never suffer infinite loss even under the log loss function. On the other hand, the following proposition shows that SVA predictors can violate the property (2) of

Table 2: The analogue of Table 1 for square loss (RMSE).

	J48 (J)				J48 Bagging (JB)				logistic regression (LR)			
	W	VA	SVA	DIR	W	VA	SVA	DIR	W	VA	SVA	DIR
Australian	0.366	0.346	0.359	0.366	0.313	0.338	0.318	0.323	0.317	0.319	0.319	0.321
Breast	0.472	0.453	0.463	0.473	0.443	0.451	0.460	0.474	0.442	0.437	0.444	0.450
Diabetes	0.449	0.431	0.443	0.449	0.407	0.415	0.420	0.427	0.399	0.401	0.401	0.402
Echo	0.478	0.456	0.460	0.482	0.427	0.417	0.423	0.444	0.457	0.443	0.446	0.475
Hepatitis	0.407	0.393	0.401	0.419	0.362	0.390	0.368	0.391	0.400	0.357	0.384	0.411
Ionosphere	0.318	0.355	0.312	0.318	0.267	0.356	0.261	0.267	0.357	0.363	0.349	0.361
Labor	0.407	0.403	0.402	0.413	0.361	0.371	0.339	0.341	0.294	0.498	0.287	0.303
Liver	0.528	0.482	0.518	0.528	0.457	0.478	0.478	0.493	0.460	0.463	0.458	0.461
Vote	0.187	0.186	0.186	0.187	0.187	0.206	0.186	0.188	0.198	0.233	0.195	0.203

	naïve Bayes (NB)				neural networks (NN)				SVM Platt (SVM)			
	W	VA	SVA	DIR	W	VA	SVA	DIR	W	VA	SVA	DIR
Australian	0.392	0.328	0.333	0.335	0.360	0.363	0.361	0.371	0.343	0.324	0.325	0.327
Breast	0.449	0.436	0.427	0.433	0.485	0.465	0.491	0.508	0.443	0.431	0.442	0.447
Diabetes	0.420	0.406	0.410	0.413	0.413	0.408	0.413	0.417	0.399	0.393	0.400	0.402
Echo	0.428	0.408	0.412	0.426	0.457	0.436	0.443	0.468	0.416	0.427	0.418	0.431
Hepatitis	0.357	0.339	0.335	0.342	0.396	0.402	0.379	0.427	0.350	0.350	0.353	0.364
Ionosphere	0.281	0.273	0.250	0.251	0.321	0.378	0.316	0.333	0.312	0.309	0.312	0.316
Labor	0.256	0.363	0.284	0.281	0.279	0.442	0.293	0.307	0.274	0.358	0.280	0.283
Liver	0.480	0.476	0.478	0.487	0.459	0.456	0.456	0.463	0.473	0.477	0.472	0.477
Vote	0.292	0.257	0.251	0.250	0.216	0.271	0.206	0.227	0.183	0.191	0.185	0.188

Algorithm 2 Simplified Venn–Abers predictor**Input:** training sequence (z_1, \dots, z_l) **Input:** test object x **Output:** multiprobabilistic prediction (p_0, p_1) **for** $y \in \{0, 1\}$ **do** set s to the scoring function for (z_1, \dots, z_l) set g_y to the isotonic calibrator for $(s(x_1), y_1), \dots, (s(x_l), y_l), (s(x), y))$ set $p_y := g_y(s(x))$ **end for**

unbiasedness in the large; in particular, they are not Venn predictors (cf. Corollary 1).

Proposition 2. *There exists a simplified Venn–Abers predictor violating (2) for some l .*

Proof. Let the object space be the real line, $\mathbf{X} := \mathbb{R}$, and the probability distribution generating independent observations (X, Y) be such that: the marginal distribution of X is continuous; the probability that $X > 0$ (and, therefore, the probability that $X < 0$) is $1/2$; the probability that $Y = 1$ given $X < 0$ is $1/3$; the probability that $Y = 1$ given $X > 0$ is $2/3$. Therefore, $\mathbb{P}(Y = 1) = 1/2$. Let l be a large number (we are using a somewhat informal language, but formalization will be obvious). Given a training set (z_1, \dots, z_l) , where $z_i = (x_i, y_i)$ for all i , the scoring function s is:

$$s(x) := \begin{cases} 0 & \text{if } x \in \{x_1, \dots, x_l\} \text{ and } x < 0 \\ 1 & \text{if } x \in \{x_1, \dots, x_l\} \text{ and } x > 0 \\ 2 & \text{if } x \notin \{x_1, \dots, x_l\}. \end{cases}$$

It is easy to see that, with high probability,

$$\underline{V} \approx 2/3, \quad \bar{V} = 1.$$

Therefore, (2) is violated. \square

Proposition 2 shows that SVA predictors are not always valid; however, the construction in its proof is artificial, and our hope is that they will be “nearly valid” in practice, since they are a modification of provably valid predictors.

For each dataset/classifier combination, we repeat the same experiment a total of 100 times for standard classifiers (denoted W in the tables), SVA, and DIR and 16 times for VA (because of the computational inefficiency of the latter) and average the results. The same 100 random splits into training and test sets are used for W, SVA, and DIR, but for VA the 16 splits are different.

Tables 1–2 compare the overall losses computed according to (12) (MLE, used in Table 1) and (13) (RMSE, used in Table 2) for probabilities generated by the standard classifiers as implemented in Weka (W) and the corresponding Venn–Abers (VA), simplified Venn–Abers (SVA), and direct isotonic-regression (DIR) predictors. The values in bold indicate the lowest of the four losses for each dataset/classifier combination. The column titles mention both fuller and shorter names for the six standard classifiers; the short name “SVM” is especially appropriate when using VA, SVA, and DIR, in which case the application of the sigmoid function in Platt’s method is redundant. The three entries of ∞ in the column W for logistic regression of Table 1 come out as infinities in our experiments only because of the limited machine accuracy: logistic regres-

sion sometimes outputs probabilistic predictions that are so close to 0 or 1 that they are rounded to 0 or 1, respectively, by hardware.

In the case of MLE, the VA and SVA methods improve the predictive performance of the majority of the standard classifiers on most datasets. A major exception is J48 Bagging. The application of bagging to J48 decision trees improves the calibration significantly as bagging involves averaging over different training sets in order to reduce the underlying classifier's instability. The application of VA and SVA to J48 Bagging is not found to improve the log or square loss significantly. What makes VA and SVA useful is that for many data sets other classifiers, less well calibrated than J48 Bagging, provide more useful scores.

In the case of RMSE, the application of VA and SVA also often improves probabilistic predictions.

Whereas in the case of square loss the DIR method often produces values comparable to VA and SVA, under log loss this method fares less well (which is not obvious from [17], which only uses square loss). In all our experiments DIR suffers infinite log loss for at least one test observation, which makes the overall MLE infinite. There are modifications of the DIR method preventing probabilistic predictions in $\{0, 1\}$ (such as those mentioned in [12], Section 3.3), but they are somewhat arbitrary.

Table 3 ranks, for each loss function and dataset, the four calibration methods: W (none), VA (Venn–Abers), SVA (simplified Venn–Abers), and DIR (direct isotonic regression). Only the first three methods are given (the best, the second best, and the second worst), where the quality of a method is measured by the performance of the best underlying classifier (indicated in parentheses using the abbreviations given in the column titles of Tables 1–2) for the given method, data set, and loss function. Notice that we are ranking the four calibration methods rather than the 24 combinations of Weka classifiers with calibration methods (e.g., were we ranking the 24 combinations, the entry for log loss and Australian would remain the same but the next entry, for log loss and Breast, would become “SVA (NB), VA (NB), VA (LR)”).

For MLE, the best algorithm is VA or SVA for 8 data sets out of 9; for RMSE this is true for 6 data sets out of 9. In all other cases the best algorithm is W rather than DIR. (And as discussed earlier, in the case of log loss the performance of DIR is especially poor.) Therefore, it appears that the most interesting comparisons are between W and VA and between W and SVA.

What is interesting is that VA and SVA perform best on equal numbers of datasets, 4 each in the case of MLE and 3 each in the case of RMSE, despite the theoretical guarantees of validity for the former method (such as Theorem 1). The similar performance of the two methods needs

to be confirmed in more extensive empirical studies, but if it is, SVA will be a preferable method because of its greater computational efficiency.

Comparing W and SVA, we can see that SVA performs better than W on 7 data sets out of 9 for MLE, and on 5 data sets out of 9 for RMSE. And comparing W and VA, we can see that VA performs better than W on 6 data sets out of 9 for MLE, and on 5 data sets out of 9 for RMSE. This suggests that SVA might be an improvement of VA not only in computational but also in predictive efficiency (but the evidence for this is very slim).

6 CONCLUSION

This paper has introduced a new class of Venn predictors thereby extending the domain of applicability of the method. Our experimental results suggest that the Venn–Abers method can potentially lead to better calibrated probabilistic predictions for a variety of datasets and standard classifiers. The method seems particularly suitable in cases where alternative probabilistic predictors produce overconfident but erroneous predictions under an unbounded loss function such as log loss. In addition, the results suggest that an alternative simplified Venn–Abers method can yield similar results while retaining computational efficiency.

Unlike the previous methods for improving the calibration of probabilistic predictors, Venn–Abers predictors enjoy theoretical guarantees of validity (shared with other Venn predictors).

Acknowledgments

Thanks to the reviewers for helpful comments, which prompted us to state explicitly Proposition 2 and add several clarifications. In our experiments in Section 5 we used the R language [14]; in particular, we used the implementation of the PAVA in the standard R package *stats* (namely, the function `isoreg`). The first author has been partially supported by EPSRC (grant EP/K033344/1).

References

- [1] Miriam Ayer, H. Daniel Brunk, George M. Ewing, W. T. Reid, and Edward Silverman. An empirical distribution function for sampling with incomplete information. *Annals of Mathematical Statistics*, 26:641–647, 1955.
- [2] Richard E. Barlow, D. J. Bartholomew, J. M. Bremner, and H. Daniel Brunk. *Statistical Inference under Order Restrictions: The Theory and Application of Isotonic Regression*. Wiley, London, 1972.
- [3] David R. Cox and David V. Hinkley. *Theoretical Statistics*. Chapman and Hall, London, 1974.

Table 3: The ranking of the best three methods (among W, VA, SVA, and DIR) for each dataset according to the two loss functions (see the text for details).

	log loss	square loss
Australian	W (JB), VA (LR), SVA (LR)	W (JB), SVA (JB), VA (LR)
Breast	SVA (NB), VA (NB), W (JB)	SVA (NB), VA (SVM), DIR (NB)
Diabetes	VA (LR), SVA (SVM), W (SVM)	VA (SVM), W (LR), SVA (SVM)
Echo	VA (SVM), SVA (NB), W (JB)	VA (NB), SVA (NB), W (SVM)
Hepatitis	VA (SVM), SVA (NB), W (JB)	SVA (NB), VA (NB), DIR (NB)
Ionosphere	SVA (NB), VA (SVM), W (SVM)	SVA (NB), DIR (NB), W (JB)
Labor	SVA (SVM), W (NN), VA (SVM)	W (NB), SVA (SVM), DIR (NB)
Liver	VA (NN), W (JB), SVA (LR)	VA (NN), SVA (NN), W (JB)
Vote	SVA (SVM), W (SVM), VA (J)	W (SVM), SVA (SVM), VA (J)

- [4] A. Philip Dawid. Probability forecasting. In Samuel Kotz, N. Balakrishnan, Campbell B. Read, Brani Vidakovic, and Norman L. Johnson, editors, *Encyclopedia of Statistical Sciences*, volume 10, pages 6445–6452. Wiley, Hoboken, NJ, second edition, 2006.
- [5] A. Frank and A. Asuncion. UCI machine learning repository, 2010.
- [6] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA data mining software: an update. *SIGKDD Explorations*, 11:10–18, 2011.
- [7] Xiaoqian Jiang, Melanie Osl, Jihoon Kim, and Lucila Ohno-Machado. Smooth isotonic regression: a new method to calibrate predictive models. *AMIA Summits on Translational Science Proceedings*, 2011:16–20, 2011.
- [8] Antonis Lambrou, Harris Papadopoulos, Ilia Nourtdinov, and Alex Gammerman. Reliable probability estimates based on support vector machines for large multiclass datasets. In Lazaros Iliadis, Ilias Maglogiannis, Harris Papadopoulos, Kostas Karatzas, and Spyros Sioutas, editors, *Proceedings of the AIAI 2012 Workshop on Conformal Prediction and its Applications*, volume 382 of *IFIP Advances in Information and Communication Technology*, pages 182–191, Berlin, 2012. Springer.
- [9] John Langford and Bianca Zadrozny. Estimating class membership probabilities using classifier learners. In Robert G. Cowell and Zoubin Ghahramani, editors, *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pages 198–205. Society for Artificial Intelligence and Statistics, 2005.
- [10] Erich L. Lehmann. *Testing Statistical Hypotheses*. Springer, New York, second edition, 1986.
- [11] Allan H. Murphy and Edward S. Epstein. Verification of probabilistic predictions: a brief review. *Journal of Applied Meteorology*, 6:748–755, 1967.
- [12] Alexandru Niculescu-Mizil and Rich Caruana. Obtaining calibrated probabilities from boosting. Technical Report arXiv:1207.1403 [cs.LG], arXiv.org e-Print archive, July 2012.
- [13] John C. Platt. Probabilities for SV machines. In Alexander J. Smola, Peter L. Bartlett, Bernhard Schölkopf, and Dale Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 2000.
- [14] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2014.
- [15] Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer, New York, 2005.
- [16] Vladimir Vovk, Glenn Shafer, and Ilia Nourtdinov. Self-calibrating probability forecasting. In Sebastian Thrun, Lawrence K. Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 1133–1140, Cambridge, MA, 2004. MIT Press.
- [17] Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In David Hand, Daniel Keim, and Raymond Ng, editors, *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 694–699, New York, 2002. ACM Press.

Tightness Results for Local Consistency Relaxations in Continuous MRFs

Yoav Wald

Amir Globerson

School of Computer Science and Engineering
The Hebrew University, Jerusalem 91904, Israel

Abstract

Finding the MAP assignment in graphical models is a challenging task that generally requires approximations. One popular approximation approach is to use linear programming relaxations that enforce local consistency. While these are commonly used for discrete variable models, they are much less understood for models with continuous variables.

Here we define local consistency relaxations of MAP for continuous pairwise Markov Random Fields (MRFs), and analyze their properties. We begin by providing a characterization of models for which this relaxation is tight. These turn out to be models that can be reparameterized as a sum of local convex functions. We also provide a simple formulation of this relaxation for Gaussian MRFs.

Next, we show how the above insights can be used to obtain optimality certificates for loopy belief propagation (LBP) in such models. Specifically, we show that the messages of LBP can be used to calculate upper and lower bounds on the MAP value, and that these bounds coincide at convergence, yielding a natural stopping criterion which was not previously available.

Finally, our results illustrate a close connection between local consistency relaxations of MAP and LBP. They demonstrate that in the continuous case, whenever LBP is provably optimal so is the local consistency relaxation.

1 INTRODUCTION

Graphical models [13] have become a key tool for describing multivariate distributions. For many models of interest, the basic inference task of finding the most likely assignment (also known as the MAP assignment) is computationally hard [26], and one must resort to approximations.

When the model variables are discrete, a popular approximation scheme is linear programming relaxations (LPR). These approximate the MAP problem via minimization of a linear function over locally consistent *pseudo-marginals*. LPRs have several advantages: they provide optimality certificates, they can be optimized via message passing algorithms, they work well in practice, and they are provably exact in some cases (e.g., binary attractive models and trees) [8, 27, 28, 36, 11].

For models with continuous variables, it is less clear how to apply the local consistency perspective of LPRs. For example, the *pseudo-marginals* now become functions rather than a discrete set of variables. Moreover, the standard consistency constraints translate into a continuum of constraints. The goal of the current paper is to study such relaxations and understand when they are tight.

Another commonly used approximate inference algorithm is loopy belief propagation (LBP)[37]. It works by passing messages along the graph, in a manner motivated by variable elimination on tree structured models. Although LBP and LPR are generally distinct algorithms, there are cases where both are exact. For example, both yield the exact MAP for tree models, maximum weight matching [3, 24] and a few other problems (see Section 8). However, there is still no general result linking LPR and LBP. Since LBP in the continuous case is fairly well understood [16, 17], we will want to link our results to known results on LBP.

We begin by defining local consistency MAP relaxations for continuous models. Technically, these will be constructed in an analogous way to the LP relaxations for discrete variables. However, they will typically not correspond to standard linear programs and thus we refer to them as local consistency relaxations (LCRs).

We obtain several surprising results on LCRs. For simplicity of presentation we focus on pairwise MRFs, but extensions to larger cliques are possible (see Section 7). Our first key result is to show that the LCR of a model is tight if the model is “convex decomposable” (CD). A model is CD if it can be expressed as a sum of pairwise convex functions

(see [17] for a similar definition).

For continuous MRFs there are several cases where LBP is known to converge to the exact MAP. It is thus of interest to relate these to our LCR results in order to further our understanding of the relation between these two approximation schemes. The relation turns out to be simple and interesting. For Gaussian MRFs LBP is provably optimal when the model is walk-summable [16, 18], a property which turns out to be equivalent to convex decomposability of the model. In other words LCR and LBP are both exact on these models. For general MRFs, LBP is provably optimal for models that are scaled diagonally dominant [17]. These turn out to be a strict subset of CD models, suggesting that either LCR is a stronger approximation or that stronger properties of LBP can be shown.

The above results on LCR and the relation to LBP lead to an important practical implication. It turns out we can use these insights to obtain runtime optimality certificates for loopy belief propagation (LBP) in a wide range of models. Specifically, we show that the messages of LBP can be used to calculate upper and lower bounds on the MAP value, and that these bounds coincide at convergence, yielding a natural stopping criterion for LBP, which was not previously available.

2 MAP IN CONTINUOUS MRFs

We begin by recalling MAP inference in graphical models and how LP relaxations are used to obtain approximate solutions of the problem.

Consider n variables X_1, \dots, X_n , and a set of singleton and pairwise functions $f_i(x_i), f_{ij}(x_i, x_j)$ where the pairs ij are a set of edges of a graph $G = (V, E)$. Use these to define a function over $\mathbf{x} = x_1, \dots, x_n$:

$$F(\mathbf{x}) = \sum_i f_i(x_i) + \sum_{ij} f_{ij}(x_i, x_j). \quad (1)$$

For now we do not assume anything about the state spaces of the variables. We refer to $F(\mathbf{x})$ as an MRF over X_1, \dots, X_n and consider the MAP problem of minimizing $F(\mathbf{x})$.¹

It will be convenient in what follows to express $F(\mathbf{x})$ as a linear function of certain functions of \mathbf{x} (our notation follows that of [33]). Assume there exists a vector of functions $\phi_i(x_i)$ of x_i , and a vector θ_i such that:

$$f_i(x_i) = \langle \theta_i, \phi_i(x_i) \rangle.$$

For example if $\phi_i(x_i) = [x_i, x_i^2]$ then $f_i(x_i)$ is the quadratic function $f_i(x_i) = \theta_{i,1}x_i + \theta_{i,2}x_i^2$. Similarly

¹We refer to this as the MAP problem since it corresponds to finding the maximum a posteriori assignment to \mathbf{x} in the model $p(\mathbf{x}) \propto \exp(-F(\mathbf{x}))$.

assume there exist functions $\phi_{i,j}(x_i, x_j)$ and vectors θ_{ij} such that:

$$f_{ij}(x_i, x_j) = \langle \theta_{ij}, \phi_{ij}(x_i, x_j) \rangle. \quad (2)$$

Denote the concatenation of all θ_i, θ_{ij} by θ and the concatenation of all ϕ_i, ϕ_{ij} functions by ϕ . Furthermore, denote the dimension of θ and ϕ by m . We can thus write:

$$F(\mathbf{x}) = \langle \theta, \phi(\mathbf{x}) \rangle, \quad (3)$$

The MAP problem has an equivalent formulation in terms of mean parameters, as shown in [33] and as reviewed next. We define the set of realizable mean parameters:

$$\mathcal{M} = \{\mu \in \mathbb{R}^m : \exists \hat{p} \in \Delta \text{ s.t. } \mathbb{E}_{\hat{p}}[\phi(\mathbf{x})] = \mu\},$$

where Δ is the set of densities over \mathbf{x} . It can be then shown that the MAP problem corresponds to optimization of a linear function over the set \mathcal{M} .

Theorem 2.1. [33] *For the MRF as defined above and the corresponding \mathcal{M} it holds that*²:

$$\min_{\mathbf{x}} F(\mathbf{x}) = \min_{\mu \in \mathcal{M}} \langle \theta, \mu \rangle. \quad (4)$$

The problem in Eq. (4) has a linear objective over m variables. m is usually not much larger than n , yet the definition of \mathcal{M} involves variables corresponding to densities and thus is generally hard to characterize explicitly. However, there are continuous variable cases where \mathcal{M} does have a compact form. For example, in Gaussian MRFs \mathcal{M} can be expressed via positive semi definiteness constraints (see [33, sec. 3.4.1]).

Finally, we recall the definition of a reparameterization of an MRF.

Definition 1. *We call any set of functions $\bar{f}_i(x_i), \bar{f}_{ij}(x_i, x_j)$ a reparameterization of $F(\mathbf{x})$ if it holds that for every \mathbf{x} :*

$$F(\mathbf{x}) = \sum_i \bar{f}_i(x_i) + \sum_{ij} \bar{f}_{ij}(x_i, x_j). \quad (5)$$

3 LOCAL CONSISTENCY RELAXATIONS

Optimizing over the set \mathcal{M} is generally hard. When X are discrete variables \mathcal{M} will involve an exponential number of inequalities. When X are continuous, even in cases where \mathcal{M} is tractable to optimize over, this optimization may be costly (e.g., a semidefinite program). This has prompted considerable research on relaxations of \mathcal{M} and the resulting optimization problem. Most of the work in this context has focused on discrete variables as reviewed next. Our goal is then to extend this framework to the continuous case.

²We note that the right hand side of Eq. (4) should have $\bar{\mathcal{M}}$ instead of \mathcal{M} . The closure is omitted for simplicity of presentation.

3.1 LCR FOR DISCRETE VARIABLES

Consider the case where X_i are discrete variables, each with D values, and the functions ϕ are defined as follows. $\phi_i(x_i)$ is D dimensional with $\phi_{i,k}(x_i) = \mathcal{I}(x_i = k)$, and $\phi_{ij}(x_i, x_j)$ is $D \times D$ dimensional with $\phi_{ij,kl}(x_i, x_j) = \mathcal{I}(x_i = k, x_j = l)$. The expected values $\mathbb{E}_{\hat{p}}[\phi_i(x_i)]$ and $\mathbb{E}_{\hat{p}}[\phi_{ij}(x_i, x_j)]$ are simply the singleton marginals $\hat{p}(x_i)$ and $\hat{p}(x_i, x_j)$ respectively. Thus, \mathcal{M} corresponds to the set of all singleton and pairwise marginals that are achieved by some distribution $\hat{p}(x)$. This is also known as the marginal polytope [33].

As mentioned earlier, the marginal polytope generally requires an exponential number of inequalities to describe. One natural alternative is to consider a *local consistency relaxation* (LCR) where instead of requiring the marginals to come from a “global” distribution $\hat{p}(x)$ we just require the singleton and pairwise marginals to be consistent. In other words, we define the set \mathcal{M}_L as the set of locally consistent pairwise marginals:

$$\mathcal{M}_L = \left\{ \mu \geq 0 : \begin{array}{l} \sum_{x_i} \mu_i(x_i) = 1 \\ \sum_{x_j} \mu_{ij}(x_i, x_j) = \mu_i(x_i) \quad \forall i, j, x_i. \end{array} \right\}$$

The local relaxation of the MAP problem is then to minimize $\mu \cdot \theta$ over $\mu \in \mathcal{M}_L$ instead of $\mu \in \mathcal{M}$. We next consider the extension of this relaxation to the continuous variable case.

3.2 LCR FOR CONTINUOUS VARIABLES

For continuous variables, a natural extension of the above is to replace the sum in the local consistency constraints with an integral:

$$\mathcal{M}_L = \left\{ \mu : \begin{array}{l} \exists \text{ densities } \hat{p}_i, \hat{p}_{ij} \text{ s.t.} \\ \int \hat{p}_{ij}(x_i, x_j) dx_j = \hat{p}_i(x_i) \quad \forall i, j, x_i \\ \mathbb{E}_{\hat{p}_i}[\phi_i(x_i)] = \mu_i \quad \forall i \\ \mathbb{E}_{\hat{p}_{ij}}[\phi_{ij}(x_i, x_j)] = \mu_{ij} \quad \forall i, j. \end{array} \right\} \quad (6)$$

In other words we consider all pairwise consistent densities, and \mathcal{M}_L are all the expected values obtained from such densities. As in the discrete case the local relaxation of MAP is then the problem:

$$\min_{\mu \in \mathcal{M}_L} \langle \theta, \mu \rangle. \quad (7)$$

Note that because $\mathcal{M} \subseteq \mathcal{M}_L$, the above minimum is a lower bound on the true MAP value (as in the case of discrete models [29]).

The key question we ask here is: for which cases is the relaxation in Eq. (7) tight? Before presenting our main result (Thm. 4.1), we will define an even looser relaxation in the next section which will be important for our analysis.

3.3 WEAK LCR AND ITS DUAL

In the constraints of Eq. (6) we require the singleton and pairwise marginals to be completely consistent. Namely, that $\hat{p}_{ij}(x_i)$, the marginal density calculated from $\hat{p}_{ij}(x_i, x_j)$ will equal $\hat{p}_i(x_i)$ for *all* x_i values. A weaker consistency constraint is to enforce that $\hat{p}_{ij}(x_j)$ and $\hat{p}_i(x_i)$ agree only on certain expected values. We next define the set corresponding to this constraint.

Given a vector of functions $\psi_i(x_i)$ for each $i \in V$,³ we define the set:

$$\mathcal{M}_L^\psi = \left\{ \mu : \begin{array}{l} \exists \text{ densities } \hat{p}_i, \hat{p}_{ij} \text{ s.t.} \\ \mathbb{E}_{\hat{p}_{ij}}[\psi_i(x_i)] = \mathbb{E}_{\hat{p}_i}[\psi_i(x_i)] \quad \forall i, j \\ \mathbb{E}_{\hat{p}_i}[\phi_i(x_i)] = \mu_i \quad \forall i \\ \mathbb{E}_{\hat{p}_{ij}}[\phi_{ij}(x_i, x_j)] = \mu_{ij} \quad \forall i, j. \end{array} \right\}$$

Thus \mathcal{M}_L^ψ enforces consistency only in the sense that $\hat{p}_{ij}(x_j)$ and $\hat{p}_i(x_i)$ agree on the expected values of ψ_i . Thus, for any choice of ψ we have $\mathcal{M}_L \subseteq \mathcal{M}_L^\psi$.

The corresponding relaxation is then defined as:

$$\min_{\mu \in \mathcal{M}_L^\psi} \langle \theta, \mu \rangle. \quad (8)$$

Basic relations between the relaxations we consider are thus summarized by the inequalities:

$$\min_{\mu \in \mathcal{M}_L^\psi} \langle \theta, \mu \rangle \leq \min_{\mu \in \mathcal{M}_L} \langle \theta, \mu \rangle \leq \min_{\mu \in \mathcal{M}} \langle \theta, \mu \rangle = \min_x F(x).$$

The dual of Eq. (8) will play an important role in subsequent sections:

Lemma 3.1. *Given a pairwise MRF with functions $f_i(x_i)$, $f_{ij}(x_i, x_j)$, the following is a dual problem of Eq. (8):*

$$\begin{aligned} \max_{\delta} \sum_i \min_{x_i} \left\{ f_i(x_i) + \sum_{j \in N(i)} \langle \delta_{ji}, \psi_i(x_i) \rangle \right\} + \\ \sum_{ij} \min_{x_i, x_j} \left\{ f_{ij}(x_i, x_j) - \langle \delta_{ji}, \psi_i(x_i) \rangle - \langle \delta_{ij}, \psi_j(x_j) \rangle \right\}. \end{aligned} \quad (9)$$

We note that the bound achieved by this dual, and hence by weak LCR, is dependent on the reparameterization $f_i(x_i)$, $f_{ij}(x_i, x_j)$ being used. The full LCR Eq. (7) may also yield different optimization problems for different reparameterizations, yet it turns out that the bound it achieves is invariant to the reparameterization. The proof of the following lemma can be found in the supplementary material.

Lemma 3.2. *LCR has the same value under any reparameterization $\{f_i(x_i), f_{ij}(x_i, x_j)\}$ of $F(x)$.*

³For example, $\psi_i(x_i)$ could be $[x_i, x_i^3]$.

4 TIGHTNESS OF LCR ON CONVEX DECOMPOSABLE MODELS

We are now ready to provide our main result on the tightness of LCRs. We first recall the definition of Convex Decomposable (CD) models, as introduced by [17]⁴ in the context of LBP analysis. Next, we show that in these models the local consistency relaxation in Eq. (7) is exact.

Definition 2. A pairwise MRF is *convex decomposable* if there exists a reparameterization $f_i(x_i), f_{ij}(x_i, x_j)$ of $F(\mathbf{x})$ such that all the functions in the reparameterization are convex.

Our result regarding tightness of LCR on CD MRFs is stated in the following theorem:

Theorem 4.1. For any CD pairwise MRF it holds that

$$\min_{\boldsymbol{\mu} \in \mathcal{M}_L} \langle \boldsymbol{\theta}, \boldsymbol{\phi}(\mathbf{x}) \rangle = \min_{\mathbf{x}} F(\mathbf{x}).$$

The proof is provided in the appendix. It relies on the following insights:

- From Lem. 3.2, the dual of weak LCR Eq. (9) taken with respect to any reparameterization of $F(\mathbf{x})$ is a lower bound on LCR's value.
- Specifically, we can consider Eq. (9) with $\boldsymbol{\psi}_i = [x_i]$ when the reparameterization $f_i(x_i), f_{ij}(x_i, x_j)$ contains convex functions.
- Assuming \mathbf{x}^* is a MAP assignment, the dual assignment $\boldsymbol{\delta}^*$ obtained by setting multipliers according to a subgradient of the pairwise functions $f_{ij}(x_i, x_j)$ taken at \mathbf{x}^* , achieves a dual objective of $F(\mathbf{x}^*)$ in Eq. (9).
- LCR's value cannot be lower than weak LCR's value, which in turn cannot be lower than $F(\mathbf{x}^*)$ because we constructed a dual assignment that achieves this objective. LCR is a lower bound on MAP, so that equality must hold.

4.1 RELATION TO LOOPY BP

In [17] it is shown that loopy BP is exact (i.e., guaranteed to converge to the true MAP assignment) on a class of models that satisfy scaled diagonal dominance. These are a strict subset of CD models. In particular, they require that a model is CD as well as its Hessian being scaled diagonally dominant. That is, there must exist a strictly positive vector $\mathbf{w} \in \mathbb{R}^n$ and a $0 < \lambda < 1$ such that for any \mathbf{x} it holds:

$$\sum_{j \in N(i)} w_j \left| \frac{\partial^2}{\partial x_i \partial x_j} F(\mathbf{x}) \right| \leq \lambda w_i \frac{\partial^2}{\partial x_i^2} F(\mathbf{x}).$$

⁴We note that the definition in [17] poses the additional demands of differentiability and that $f_i(x_i)$ are strictly convex, so the class we define here is a larger one.

Thus, we conclude that given currently known exactness results on LBP, the LCR approximation is stronger in the sense that it is exact whenever LBP is known to be exact. It remains an open question whether it can be shown that LBP is exact on CD models.

5 GAUSSIAN MRFs AND THEIR RELAXATION

Gaussian MRFs (GMRFs) have been studied widely, and are also of practical interest [5, 6, 16, 18, 34]. Here we give a simple characterization of \mathcal{M}_L for GMRFs, and then discuss when the corresponding relaxations are tight. We give a stronger result than Theorem 4.1 by showing that LCR is tight if and only if the Gaussian model is CD. Specifically, we show that for non CD models the LCR optimization problem has a value of $-\infty$, and is thus a meaningless relaxation.

Recall that a GMRF $F(\mathbf{x})$ is a quadratic form:

$$F(\mathbf{x}) = \frac{1}{2} \mathbf{x}^\top \Gamma \mathbf{x} - \mathbf{h}^\top \mathbf{x},$$

where $\Gamma \geq 0$. The function vectors $\boldsymbol{\phi}_i(x_i), \boldsymbol{\phi}_{ij}(x_i, x_j)$ for this type of MRF are given by

$$\boldsymbol{\phi}_i(x_i) = \{x_i, x_i^2\}, \boldsymbol{\phi}_{ij}(x_i, x_j) = \{x_i x_j\}. \quad (10)$$

As stated in [33], the set of realizable mean parameters \mathcal{M} in this case is the set of all first and second moments that can be realized by a density \hat{p} . Namely:

$$\mathcal{M} = \{(\Sigma, \boldsymbol{\eta}) : \Sigma - \boldsymbol{\eta} \boldsymbol{\eta}^\top \geq 0\}. \quad (11)$$

Here the elements Σ_{ij} correspond to the expected values of $x_i x_j$, the diagonal elements Σ_{ii} are the expected values of x_i^2 and $\boldsymbol{\eta}$ are the expected values of x_i .

5.1 A CHARACTERIZATION OF \mathcal{M}_L FOR GMRFs

Given this simple form of \mathcal{M} , it is interesting to see what \mathcal{M}_L translates into for this case. To characterize \mathcal{M}_L we define:

Definition 3. Given an $n \times n$ matrix A , and an edge $(i, j) \in E$, define the following sub matrix of A

$$A_{[ij]} = \begin{bmatrix} A_{ii} & A_{ij} \\ A_{ji} & A_{jj} \end{bmatrix}.$$

Similarly, for an n dimensional vector v , define

$$v_{[ij]} = \begin{bmatrix} v_i \\ v_j \end{bmatrix}.$$

Claim 5.1. For GMRFs it holds that

$$\mathcal{M}_L = \{(\Sigma, \boldsymbol{\eta}) : \Sigma_{[ij]} - \boldsymbol{\eta}_{[ij]} \boldsymbol{\eta}_{[ij]}^\top \geq 0 \ \forall ij \in E\}. \quad (12)$$

The proof can be found in the appendix. The characterization is very intuitive: it says that Σ should be such that its pairwise submatrices are valid covariance matrices of two variables with mean given by η .

5.1.1 LCR is Unbounded for non-CD GMRFs

We now turn to give a full characterization of the GMRFs on which LCR is tight. From Thm. 4.1 we know that whenever a GMRF is convex decomposable then LCR is tight, yet we do not know if tightness holds for any other GMRFs. It turns out that it does not, and in fact the objective of LCR is unbounded for non CD models. The proof relies on the following two claims. First we claim that for GMRFs, under a certain choice of $\psi_i(x_i)$, \mathcal{M}_L^ψ as given for the weak LCR in Eq. (8) is equal to \mathcal{M}_L :

Claim 5.2. *Let \mathcal{M}_L be the set defined in Eq. (6) with the functions in Eq. (10). For $\psi_i(x_i) = [x_i, x_i^2]$ it holds that:*

$$\mathcal{M}_L = \mathcal{M}_L^\psi. \quad (13)$$

From Claim 5.2 we gather that weak LCR achieves the same bound as LCR, and that Eq. (9) is also a dual of LCR. The second claim then states that Eq. (9) is unbounded when the GMRF is not CD.

Claim 5.3. *For any non-CD GMRF, the value of Eq. (9) taken with $\psi_i(x_i) = [x_i, x_i^2]$ is $-\infty$ for any choice of δ .*

The proof of both claims is provided in the appendix. It is easy to check that Slater's conditions hold for LCR as defined by Eq. (12) (see e.g. [4, p. 523] for the conditions). Thus when the optimum of LCR is bounded (i.e. larger than $-\infty$), strong duality holds and Eq. (9) obtains the same optimal value. Claim 5.3 states that this does not happen for non-CD GMRFs. We have thus shown the following corollary:

Corollary 5.4. *LCR is unbounded on any non-CD GMRF.*

5.2 RELATION TO LOOPY BP

Several works have analyzed the properties of LBP when applied to GMRFs [16, 34, 18]. These show that when a model is CD then LBP is exact.⁵ Note that the definition of convex decomposability used in [18] is somewhat more restrictive than what we consider here. It requires the pairwise functions $f_{ij}(x_i, x_j)$ in the convex decomposition to be strictly convex.

Given our results above, we conclude that LCR is exact precisely on the models where LBP is known to be exact. That is with the subtle exception of cases where the pairwise functions in the convex decomposition are not strictly convex, and then LBP is not known to be exact.

⁵As stated in Section 1, there are other characterizations of these models, e.g. walk summability, which turn out to be equivalent to convex decomposability.

6 A STOPPING CRITERION FOR LBP

In this section we highlight a practical application resulting from the close LCR and LBP connection. Since LBP does not optimize a clearly defined objective, it is hard to monitor its convergence. Below we show how in many cases, simple upper and lower bounds can be calculated for LBP.

Consider an MRF where f_{ij}, f_i are convex and LBP is known to be exact. We will provide upper and lower bounds on the MAP objective in this case and show that they are tight at convergence. These bounds can thus be used as a reliable, easy to apply stopping criterion for LBP in these cases.⁶

Given our assumption on the MRF, it follows that it is CD. Thus, the LCR is tight, and the maximum of the dual objective Eq. (9) will equal the MAP value. This leads us to the following bounding scheme. At each iteration of BP, calculate an estimate of the MAP and denote it by \mathbf{x}^t . Clearly $F(\mathbf{x}^t)$ is an upper bound on the MAP value.

The procedure for obtaining the lower bound is more involved, although technically simple. It is described (along with the upper bound) in Algorithm 1. First, it is easy to see that d in Algorithm 1 is a lower bound since it is a value of the dual in Eq. (9). Second, this bound is tight when \mathbf{x}^t is the MAP assignment, as the following result states:

Lemma 6.1. *Assume $\mathbf{x}^t \in \arg \min_{\mathbf{x}} F(\mathbf{x})$, then the bound returned by the procedure satisfies $p = d$.*

Algorithm 1 Bounding Scheme for Sum of Convex Functions

Input: MAP estimate \mathbf{x}^t at time t , accuracy parameter ε , convex reparameterization $\{f_i(x_i), f_{ij}(x_i, x_j)\}$.
For each $ij \in E$, find $g \in \partial f_{ij}(x_i^t, x_j^t)$ and set

$$\hat{\delta}_{ji} = g_i, \hat{\delta}_{ij} = g_j.$$

Set for each $i \in V$ and $ij \in E$

$$\bar{f}_i(x_i) = f_i(x_i) + \sum_{j \in N(i)} \hat{\delta}_{ji} x_i$$

$$\bar{f}_{ij}(x_i, x_j) = f_{ij}(x_i, x_j) - \hat{\delta}_{ji} x_i - \hat{\delta}_{ij} x_j$$

Calculate primal and dual objectives ⁷

$$p = \sum_i f_i(x_i^t) + \sum_{ij} f_{ij}(x_i^t, x_j^t)$$

$$d = \sum_i \min_{x_i} \bar{f}_i(x_i) + \sum_{ij} \min_{x_i, x_j} \bar{f}_{ij}(x_i, x_j)$$

If $p - d < \varepsilon$ then determine convergence.

⁶It is possible to apply these bounds to any iterative algorithm that provides a sequence of approximate MAP assignments. We focus on LBP since the bounds are tight in the case we address.

The proof of this lemma follows exactly the same argument given in the third step for the proof of Thm. 4.1. When the singleton functions f_i are strongly convex and the objective is smooth, then it is also possible to give a guarantee on the size of the bound for x^t that is not a MAP assignment (see Lem. B.1 in the supplementary material).

Thus, we have obtained a scheme that provides tight lower and upper bounds on the iterates of BP in a subclass of the cases when it converges. Figure 1 illustrates these bounds for a Gaussian MRF.

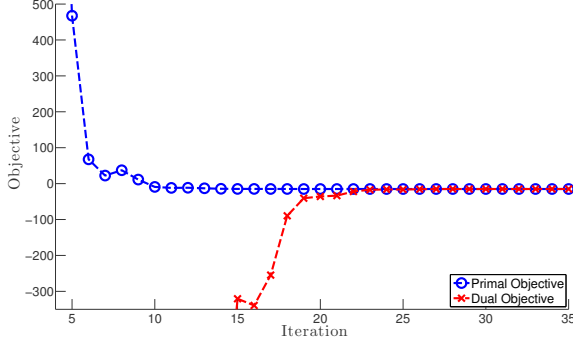


Figure 1: Illustration of the bounds for a 10×10 grid Gaussian MRF. We generate random convex quadratic functions for each node and edge in the model, and use the described bounding scheme to determine LBP’s convergence.

7 HIGHER ORDER MODELS

Thus far we only considered pairwise models. In this section we briefly mention the generalization of our results to higher order models. We propose two possible LCR approaches in this context and provide the related tightness results.

A non pairwise MRF may be written as follows:

$$F(\mathbf{x}) = \sum_{\alpha \in C} f_{\alpha}(\mathbf{x}_{\alpha}),$$

where $C \subseteq 2^V$. One option to define \mathcal{M}_L for this case is:

$$\mathcal{M}_L = \left\{ \mu : \begin{array}{ll} \exists \text{ densities } \hat{p}_i, \hat{p}_{\alpha} \text{ s.t.} & \\ \int \hat{p}_{\alpha}(\mathbf{x}_{\alpha}) d\mathbf{x}_{\alpha \setminus i} = \hat{p}_i(x_i) & \forall \alpha, i \in \alpha, x_i \\ \mathbb{E}_{\hat{p}_{\alpha}}[\phi_{\alpha}(\mathbf{x}_{\alpha})] = \mu_{\alpha} & \forall \alpha \in C. \end{array} \right\} \quad (14)$$

In other words, \mathcal{M}_L constrains marginals over the α sets to agree on singleton marginals. Another reasonable choice is to define a set \mathcal{M}_C that enforces stronger consistency

⁷Notice that in order to calculate d it is not required to perform optimization over $\bar{f}_{ij}(x_i, x_j)$. The manner in which we set \hat{d} guarantees that the minimum is attained at (x_i^t, x_j^t) . The calculation of d will only demand optimizing $\bar{f}_i(x_i)$ for each i .

constraints than \mathcal{M}_L :

$$\int \hat{p}_{\alpha}(\mathbf{x}_{\alpha}) d\mathbf{x}_{\alpha \setminus \beta} = \int \hat{p}_{\beta}(\mathbf{x}_{\beta}) d\mathbf{x}_{\beta \setminus \alpha} \quad \forall \alpha, \beta \in C. \quad (15)$$

Instead of consistency over single variables, \mathcal{M}_C enforces consistency on the overlap of pairs $\alpha, \beta \in C$.

Similar derivations to those of the pairwise case lead to tightness characterization for the constraints above. For our tightness results to carry, $F(\mathbf{x})$ should be given as a sum of convex functions. The first result states that if the functions $f_{\alpha}(\mathbf{x}_{\alpha})$ are all convex, then the \mathcal{M}_L relaxation is tight.

Claim 7.1. *If $F(\mathbf{x})$ is given as a sum of convex functions then $\min_{\mu \in \mathcal{M}_L} \langle \theta, \mu \rangle = \min_{\mathbf{x}} F(\mathbf{x})$.*

In this case we can also use the bounds described in Section 6. An interesting example of such a model is the one underlying the AMP algorithm of Donoho et al. [19].

The relaxation defined by Eq. (15) may be more complicated to solve due to the additional constraints. At the cost of this complication, the obtained relaxation is invariant to reparameterizations and an analogue of Thm. 4.1 holds.⁸

Definition 4. *An MRF is CD w.r.t $C \subseteq 2^V$ if there exists a reparameterization $\{\bar{f}_{\alpha}(\mathbf{x}_{\alpha})\}$ for $F(\mathbf{x})$ such that all the functions in the reparameterization are convex.*

Claim 7.2. *For any MRF that is CD w.r.t C it holds that $\min_{\mu \in \mathcal{M}_C} \langle \theta, \mu \rangle = \min_{\mathbf{x}} F(\mathbf{x})$.*

See supplementary material for proofs of the above results.

8 RELATED WORK

The current paper studies local consistency relaxations as applied to continuous MRFs, and their relation to loopy belief propagation. Below we briefly survey related results on this relation, focusing on discrete variable models where local consistency relaxations are typically considered.

For discrete tree structured MRFs, it can be shown that LBP and LCR are equivalent in the sense that they are both tight, and in fact there is a mapping between dual LCR variables and BP optima (see [33, p. 200]).

In [35] Weiss *et al.* consider the relation between LCR and convex belief propagation (and *not* standard LBP). They show that if a convex variant of max-product LBP converges, and the sharpened version of its beliefs (where sharpening means to distribute all probability mass evenly between maximizing arguments of the beliefs) are locally consistent, then they are a solution to the LCR.

For maximum weight bipartite matching it is known that LBP is exact [3] and so is the standard LP relaxation of the problem [31] (see also [1, sec. 6] on the relation between

⁸This result holds under a non-restrictive assumption that $\alpha \cap \beta \in C$ for all $\alpha, \beta \in C$ and also $\{i\} \in C$ for all $i \in V$.

this relaxation and local consistency relaxations). For non bipartite matching a more subtle result is available [24] showing that if the LCR does not have fractional optima then LBP is exact. Otherwise LBP will not be exact. Related results are available for maximum weight independent set [25] and packing and covering problems [7].

For the problem of decoding LDPC codes, both LCR approaches [9] and LBP [21] have been successful. Although some relations between these have been shown [32, 2], a clear link establishing cases where they are both exact has not been provided.

Tarlow *et al.* [30] show that for the min-cut problem, where LCR is known to be tight (e.g., see [31]), a modified version of LBP is exact as well.

Another connection between the LPR and message passing algorithms relies on the notion of graph covers [23, 12]. It is shown that for discrete models, LPR solves the MAP problem on a model that has the lowest objective amongst a family of graphical models who are in some sense isomorphic to the model it tries to approximate. Message passing algorithms are unable to distinguish between these isomorphic models, and thus an intuitive link between LPR and message passing is made.

On GMRFs, both LBP [16, 34, 18] and graph covers [22] were studied. One of the conclusions reached by studying graph covers [22] is that for non-CD GMRFs, dual coordinate ascent algorithms [10, 14, 15] are bound to fail at producing the MAP estimate. Our result for GMRFs provides an arguably simpler explanation of this failure: These algorithms perform dual coordinate ascent on the LCR, and whenever the LCR is inapplicable (as in the case of non-CD GMRFs) they yield a useless bound. See Section 5.2 for further discussion of GMRFs and LBP.

Relaxations for continuous MRFs have been less studied. The most relevant work is [20], who arrived at the dual of Eq. (7) (notice we did not use this dual, rather we used Eq. (9) which is a dual of Eq. (8)) but did not analyze when these are exact. Another recent work [38] suggested a different relaxation for continuous variables, but one that is more involved than standard LPRs and also has no clear connection to BP.

Another work on MAP estimation in continuous MRFs suggested an algorithm called Linear Coordinate Descent [40]. This algorithm was later generalized by the authors [39]. For both versions, the authors do not give an objective over which the updates yield an improvement at each step. In fact, our analysis provides a very simple interpretation of the algorithm in [40]. It essentially performs dual coordinate ascent on the dual of weak LCR (see Eq. (9)). This also leads to a much simpler convergence proof than that given in [40], since coordinate ascent in this case has a unique maximum and therefore converges globally [4].

9 DISCUSSION

We considered the MAP problem for MRFs on continuous variables, and derived a local relaxation for these. For convex decomposable MRFs we showed that these relaxations are in fact exact. For Gaussian MRFs we provided a stronger result showing that convex decomposability is necessary and sufficient for exactness of local relaxations.

Comparing our results to those on exactness of loopy belief propagation we find that local relaxations are exact in a strictly larger class of models. This further strengthens the known relation between LBP and LCR, adding to numerous other models where exactness of local relaxations coincides with exactness of LBP. It also leaves interesting open questions. For example, does BP in fact converge on general CD models or is the scaled diagonal dominance condition of [17] necessary.

We note that it is possible to use the LCR and its dual to derive coordinate ascent algorithms for the dual LCR. We have indeed derived MPLP-like algorithms [27] for this case. However, our experiments (not shown here) indicate that they are typically slower than LBP. Thus, LBP remains an attractive option for optimizing such models, and the tight bounds we develop in Section 6 are very useful when using it in practice. An interesting open problem is to devise such bounds for LBP in other models, and to provide general transformations between LCR and LBP solutions.

A Proofs

A.1 Proof of Lem. 3.1

Proof. Let us write \mathcal{M}_L^ψ with auxiliary variables η :

$$\mathcal{M}_L^\psi = \left\{ \mu, \eta : \begin{array}{ll} \exists \hat{p}_i, \hat{p}_{ij} \text{ s.t.} & \\ \mathbb{E}_{\hat{p}_i} [\phi_i(x_i)] = \mu_i & \forall i \\ \mathbb{E}_{\hat{p}_{ij}} [\phi_{ij}(x_i, x_j)] = \mu_{ij} & \forall ij \\ \mathbb{E}_{\hat{p}_i} [\psi_i(x_i)] = \eta_i & \forall i \\ \mathbb{E}_{\hat{p}_{ij}} [\psi_i(x_i)] = \eta_{ji} & \forall i, j \in N(i) \\ \eta_{ji} = \eta_i & \forall i, j \in N(i). \end{array} \right\}$$

Define for each $i \in V, ij \in E$ the sets of realizable mean parameters:

$$\mathcal{M}_i = \left\{ \mu_i, \eta_i : \begin{array}{l} \exists \hat{p}_i \text{ s.t.} \\ \mathbb{E}_{\hat{p}_i} [\phi_i(x_i)] = \mu_i \\ \mathbb{E}_{\hat{p}_i} [\psi_i(x_i)] = \eta_i \end{array} \right\},$$

$$\mathcal{M}_{ij} = \left\{ \mu_{ij}, \eta_{ij}, \eta_{ji} : \begin{array}{l} \exists \hat{p}_{ij} \text{ s.t.} \\ \mathbb{E}_{\hat{p}_{ij}} [\phi_{ij}(x_i, x_j)] = \mu_{ij} \\ \mathbb{E}_{\hat{p}_{ij}} [\psi_i(x_i)] = \eta_{ji} \\ \mathbb{E}_{\hat{p}_{ij}} [\psi_j(x_j)] = \eta_{ij}. \end{array} \right\}$$

Then \mathcal{M}_L^ψ can now be written compactly as:

$$\mathcal{M}_L^\psi = \left\{ \mu, \eta : \begin{array}{ll} (\mu_i, \eta_i) \in \mathcal{M}_i & \forall i \\ (\mu_{ij}, \eta_{ij}, \eta_{ji}) \in \mathcal{M}_{ij} & \forall ij \\ \eta_{ji} = \eta_i & \forall i, j \in N(i). \end{array} \right\}$$

Using the expressions for $f_i(x_i), f_{ij}(x_i, x_j)$ from Eq. (2), our problem is:

$$\min_{(\mu, \eta) \in \mathcal{M}_L^\psi} \sum_i \langle \theta_i, \mu_i \rangle + \sum_{ij} \langle \theta_{ij}, \mu_{ij} \rangle$$

We now assign a Lagrange multiplier to each consistency constraint

$$\begin{aligned} \delta_{ji} &\leftrightarrow \eta_{ji} = \eta_i \\ \delta_{ij} &\leftrightarrow \eta_{ij} = \eta_j, \end{aligned}$$

and write the resulting Lagrangian:

$$\begin{aligned} \mathcal{L}(\delta, \eta, \mu) = & \sum_i \left\{ \langle \theta_i, \mu_i \rangle + \sum_{j \in N(i)} \langle \delta_{ji}, \eta_i \rangle \right\} \\ & + \sum_{ij} \left\{ \langle \theta_{ij}, \mu_{ij} \rangle - \langle \delta_{ij}, \eta_{ij} \rangle - \langle \delta_{ji}, \eta_{ji} \rangle \right\}. \end{aligned}$$

To obtain a dual, we first minimize $\mathcal{L}(\delta, \mu, \eta)$ w.r.t μ, η under the remaining constraints: $(\mu_i, \eta_i) \in \mathcal{M}_i, (\mu_{ij}, \eta_{ij}, \eta_{ji}) \in \mathcal{M}_{ij}$. Since the relevant variables for each constraint all lie in a single summand, we can push the minimization inside the sums:

$$\begin{aligned} \mathcal{L}(\delta) = & \sum_i \min_{\mu_i, \eta_i \in \mathcal{M}_i} \left\{ \langle \theta_i, \mu_i \rangle + \sum_{j \in N(i)} \langle \delta_{ji}, \eta_i \rangle \right\} + \\ & \sum_{ij} \min_{\mu_{ij}, \eta_{ij}, \eta_{ji} \in \mathcal{M}_{ij}} \left\{ \langle \theta_{ij}, \mu_{ij} \rangle - \langle \delta_{ij}, \eta_{ij} \rangle - \langle \delta_{ji}, \eta_{ji} \rangle \right\}. \end{aligned}$$

Each summand includes optimization over the set of realizable mean parameters, thus according to Thm. 2.1 our Lagrangian is given by:

$$\begin{aligned} \mathcal{L}(\delta) = & \sum_i \min_{x_i} \left\{ \langle \theta_i, \phi_i(x_i) \rangle + \sum_{j \in N(i)} \langle \delta_{ji}, \psi_i(x_i) \rangle \right\} + \\ & \sum_{ij} \min_{x_i, x_j} \left\{ \langle \theta_{ij}, \phi_{ij}(x_i, x_j) \rangle \right. \\ & \left. - \langle \delta_{ij}, \phi_{ij}(x_j) \rangle - \langle \delta_{ji}, \phi_{ji}(x_i) \rangle \right\}. \end{aligned}$$

The desired dual is obtained by maximizing over δ . \square

A.2 Proof of Thm. 4.1

Proof. Assume $\{f_i(x_i), f_{ij}(x_i, x_j)\}$ is a reparameterization of $F(x)$ for which all the functions are convex. According to Lem. 3.2 the minimum of LCR does not depend

on the reparameterization, thus it is enough to prove weak LCR's tightness with respect to this decomposition in order to establish LCR's tightness.

Consider weak LCR taken with $\psi_i(x_i) = [x_i]$. In this case the dual problem Eq. (9) is:

$$\begin{aligned} \max_{\delta} \sum_i \min_{x_i} \left\{ f_i(x_i) + \sum_{j \in N(i)} \delta_{ji} x_i \right\} + \\ \sum_{ij} \min_{x_i, x_j} \left\{ f_{ij}(x_i, x_j) - \delta_{ji} x_i - \delta_{ij} x_j \right\}. \end{aligned}$$

Let $x^* \in \arg \min_x F(x)$, and for each $ij \in E$ take some $g \in \partial f_{ij}(x_i^*, x_j^*)$. Set the multipliers δ_{ji}, δ_{ij} as the components of g :

$$\delta_{ji}^* = g_i, \delta_{ij}^* = g_j,$$

and define the reparameterization obtained under δ^* :

$$\begin{aligned} \bar{f}_i(x_i) &= f_i(x_i) + \sum_{j \in N(i)} \delta_{ji}^* x_i \\ \bar{f}_{ij}(x_i, x_j) &= f_{ij}(x_i, x_j) - \delta_{ji}^* x_i - \delta_{ij}^* x_j. \end{aligned}$$

Each of the functions \bar{f}_i, \bar{f}_{ij} is a sum convex functions, hence is convex. From how we set g , each \bar{f}_{ij} is minimized at (x_i^*, x_j^*) . We also have:

$$\sum_i \bar{f}_i(x_i^*) = F(x^*) - \sum_{ij} \bar{f}_{ij}(x_i^*, x_j^*).$$

It holds that $0 \in \partial \bar{f}_{ij}(x_i^*, x_j^*)$ and that $0 \in \partial F(x^*)$ (because x^* is a minimizer). From additivity of the subgradient, we now get $0 \in \partial \sum_i \bar{f}_i(x_i^*)$, and since each function in the sum depends on a different variable it also holds that $0 \in \partial \bar{f}_i(x_i^*)$ for each i . We conclude that x^* minimizes each function in the reparameterization \bar{f}_i, \bar{f}_{ij} , and the dual objective our assignment δ^* achieves is:

$$\sum_i \bar{f}_i(x_i^*) + \sum_{ij} \bar{f}_{ij}(x_i^*, x_j^*) = F(x^*).$$

Any objective reached by a feasible dual assignment yields a lower bound on LCR's optimal objective, thus LCR's optimum is bounded below the MAP objective and we may conclude that the relaxation is tight. \square

A.3 Proof of Claims on LCR for GMRFs

A.3.1 Proof of Claims 5.1 and 5.2

Recall that for a GMRF we have:

$$\phi_i = \{x_i, x_i^2\}, \phi_{ij} = \{x_i x_j\}.$$

For any feasible element in \mathcal{M}_L , let \hat{p}_i, \hat{p}_{ij} be the densities that generated these feasible mean parameters. For any $ij \in E$

$E, i \in V$ denote:

$$\begin{aligned}\Sigma_{ij} &= \mathbb{E}_{\hat{p}_{ij}} [x_i x_j], \\ \Sigma_{ii} &= \mathbb{E}_{\hat{p}_{ij}} [x_i^2] = \mathbb{E}_{\hat{p}_i} [x_i^2], \\ \eta_i &= \mathbb{E}_{\hat{p}_{ij}} [x_i] = \mathbb{E}_{\hat{p}_i} [x_i].\end{aligned}\quad (16)$$

The equalities of expectations with respect to \hat{p}_i and \hat{p}_{ij} hold due to local consistency constraints. Since for any $ij \in E$, the parameters $(\Sigma_{[ij]}, \eta_{[ij]})$ are first and second moments of the density \hat{p}_{ij} then it must hold that:

$$\Sigma_{[ij]} - \eta_{[ij]} \eta_{[ij]}^\top \geq 0. \quad (17)$$

On the other hand, assume we are given (Σ, η) such that Eq. (17) holds for all $ij \in E$. Take \hat{p}_{ij} as the bivariate Gaussian density with moments $(\Sigma_{[ij]}, \eta_{[ij]})$ for all $ij \in E$, and \hat{p}_i as the univariate Gaussian density with moments (Σ_{ii}, η_i) for all $i \in V$. These densities satisfy the local consistency constraints:

$$\int \hat{p}_{ij}(x_i, x_j) dx_j = \hat{p}_i(x_i),$$

which means $(\Sigma, \eta) \in \mathcal{M}_L$ and that Claim 5.1 holds.

To see Claim 5.2 holds, we use the same type of argument and prove $\mathcal{M}_L^\psi \subseteq \mathcal{M}_L$. Given any feasible element in \mathcal{M}_L^ψ , consider the densities \hat{p}_i, \hat{p}_{ij} who generated it. The weak local consistency constraints imposed by \mathcal{M}_L^ψ assure that the equalities in Eq. (16) hold. Thus Eq. (17) also holds and this element must also be feasible in \mathcal{M}_L .

A.3.2 Proof of Claim 5.3

Proof. Let δ be any assignment to the dual Eq. (9), we will prove it achieves an objective of $-\infty$. It will then follow that this is the maximal value achieved by the dual, and in turn the minimal value achieved by LCR. Let us define the functions:

$$\begin{aligned}\bar{f}_i(x_i) &= f_i(x_i) + \sum_{j \in N(i)} \langle \delta_{ji}, \psi_i(x_i) \rangle \\ \bar{f}_{ij}(x_i, x_j) &= f_{ij}(x_i, x_j) - \langle \delta_{ji}, \psi_i(x_i) \rangle - \langle \delta_{ij}, \psi_j(x_j) \rangle.\end{aligned}$$

The dual objective may then be written as:

$$\sum_i \min_{x_i} \bar{f}_i(x_i) + \sum_{ij} \min_{x_i, x_j} \bar{f}_{ij}(x_i, x_j).$$

Under the choice of $\psi_i(x_i)$ made in Claim 5.2, each function in $\{\bar{f}_i(x_i), \bar{f}_{ij}(x_i, x_j)\}$ is quadratic. $\{\bar{f}_i(x_i), \bar{f}_{ij}(x_i, x_j)\}$ is also a reparameterization of $F(x)$, so for non-CD GMRFs, at least one of these functions must be non-convex. The minimum of a non-convex quadratic function is unbounded, so one of the minimizations in the sum of the dual objective must be unbounded, and the objective achieved by δ is $-\infty$. \square

Acknowledgements

This work was supported by the ISF Centers of Excellence grant 1789/11. We would like to thank Nir Rosenfeld for his useful comments on this paper.

References

- [1] C. Arora and A. Globerson. Higher order matching for consistent multiple target tracking. In *ICCV*, 2013.
- [2] S. Arora, C. Daskalakis, and D. Steurer. Message passing algorithms and improved lp decoding. In *STOC*, pages 3–12, 2009.
- [3] M. Bayati, D. Shah, and M. Sharma. Max-product for maximum weight matching: Convergence, correctness, and lp duality. *IEEE Trans. on Inf. Theory*, 54(3):1241–1251, 2008.
- [4] D. P. Bertsekas. *Nonlinear programming*. Athena Scientific, 1999.
- [5] D. Bickson. *Gaussian Belief Propagation: Theory and Application*. PhD thesis, Hebrew University of Jerusalem, 2008.
- [6] B. Cseke and T. Heskes. Properties of bethe free energies and message passing in gaussian models. *Journal of Artificial Intelligence Research*, 41(2):1–24, 2011.
- [7] G. Even and N. Halabi. Message-passing algorithms for packing and covering linear programs with zero-one matrices. *arXiv preprint arXiv:1302.3518*, 2013.
- [8] J. Feldman, T. Malkin, R. A. Servedio, C. Stein, and M. J. Wainwright. LP decoding corrects a constant fraction of errors. *IEEE Trans. on Inf. Theory*, 53(1):82–89, 2007.
- [9] J. Feldman, M. J. Wainwright, and D. R. Karger. Using linear programming to decode binary linear codes. *IEEE Trans. on Inf. Theory*, 51(3):954–972, 2005.
- [10] A. Globerson and T. Jaakkola. Fixing max-product: Convergent message passing algorithms for map lp-relaxations. *Advances in Neural Information Processing Systems*, 21(1.6), 2007.
- [11] J. K. Johnson. *Convex relaxation methods for graphical models: Lagrangian and maximum entropy approaches*. PhD thesis, Massachusetts Institute of Technology, 2008.
- [12] R. Koetter and P. O. Vontobel. Graph-covers and iterative decoding of finite length codes. In *Proc. Int. Symp. on Turbo Codes and Related Topics*, 2003.
- [13] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

- [14] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Trans. on Pattern Anal. and Mach. Int.*, 28(10):1568–1583, 2006.
- [15] V. Kovalevsky and V. Koval. A diffusion algorithm for decreasing energy of max-sum labeling problem. *Glushkov Institute of Cybernetics, Kiev, USSR*, 1975.
- [16] D. Malioutov, J. Johnson, and A. Willsky. Walk-sums and belief propagation in gaussian graphical models. *The Journal of Machine Learning Research*, 7:2031–2064, 2006.
- [17] C. Moallemi and B. Van Roy. Convergence of the min-sum algorithm for convex optimization. In *Proc. of the 45th Allerton Conference on Communication, Control and Computing*, 2007.
- [18] C. Moallemi and B. Van Roy. Convergence of min-sum message passing for quadratic optimization. *IEEE Trans. on Inf. Theory*, 55(5):2413–2423, 2009.
- [19] A. Montanari. Graphical models concepts in compressed sensing. *Compressed Sensing: Theory and Applications*, pages 394–438, 2012.
- [20] J. Peng, T. Hazan, D. McAllester, and R. Urtasun. Convex max-product algorithms for continuous mrfs with applications to protein folding. In *Proc. ICML*, 2011.
- [21] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke. Design of capacity-approaching irregular low-density parity-check codes. *IEEE Trans. on Inf. Theory*, 47(2):619–637, 2001.
- [22] N. Ruozzi and S. Tatikonda. Message-passing algorithms for quadratic minimization. *The Journal of Machine Learning Research*, 14(1):2287–2314, 2013.
- [23] N. Ruozzi and S. Tatikonda. Message-passing algorithms: Reparameterizations and splittings. *IEEE Trans. on Inf. Theory*, 59(9):5860–5881, 2013.
- [24] S. Sanghavi, D. M. Malioutov, and A. S. Willsky. Linear programming analysis of loopy belief propagation for weighted matching. In *NIPS*, 2007.
- [25] S. Sanghavi, D. Shah, and A. Willsky. Message passing for maximum weight independent set. *IEEE Trans. on Inf. Theory*, 55(11):4822–4834, 2009.
- [26] S. E. Shimony. Finding maps for belief networks is NP-hard. *Artificial Intell.*, 68(2):399–410, 1994.
- [27] D. Sontag, A. Globerson, and T. Jaakkola. Introduction to dual decomposition for inference. In S. Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*, pages 219–254. MIT Press, 2011.
- [28] D. Sontag, T. Meltzer, A. Globerson, Y. Weiss, and T. Jaakkola. Tightening LP relaxations for MAP using message-passing. In *Uncertainty in Artificial Intelligence*, pages 503–510. AUAI Press, 2008.
- [29] D. A. Sontag. *Approximate inference in graphical models using LP relaxations*. PhD thesis, Massachusetts Institute of Technology, 2010.
- [30] D. Tarlow, I. E. Givoni, R. S. Zemel, and B. J. Frey. Graph cuts is a max-product algorithm. In *UAI*, pages 671–680, 2011.
- [31] B. Taskar, S. Lacoste-Julien, and M. Jordan. Structured prediction, dual extragradient and Bregman projections. *Journal of Machine Learning Research*, pages 1627–1653, 2006.
- [32] P. Vontobel and R. Koetter. On the relationship between linear programming decoding and min-sum algorithm decoding. In *ISITA*, pages 991–996, 2004.
- [33] M. Wainwright and M. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.
- [34] Y. Weiss and W. Freeman. Correctness of belief propagation in Gaussian graphical models of arbitrary topology. *Neural computation*, 13(10):2173–2200, 2001.
- [35] Y. Weiss, C. Yanover, and T. Meltzer. Map estimation, linear programming and belief propagation with convex free energies. In *UAI*, 2007.
- [36] C. Yanover, O. Schueler-Furman, and Y. Weiss. Minimizing and learning energy functions for side-chain prediction. *Journal of Computational Biology*, 15(7):899–911, 2008.
- [37] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. *Exploring artificial intelligence in the new millennium*, 8:236–239, 2003.
- [38] C. Zach and P. Kohli. A convex discrete-continuous approach for markov random fields. *Computer Vision–ECCV 2012*, 2012.
- [39] G. Zhang and R. Heusdens. Generalized linear coordinate-descent message-passing for convex optimization. In *ICASSP*, pages 2009–2012. IEEE, 2012.
- [40] G. Zhang and R. Heusdens. Linear coordinate-descent message passing for quadratic optimization. *Neural computation*, 24(12):3340–3370, 2012.

Bayesian Filtering with Online Gaussian Process Latent Variable Models

Yali Wang
Laval University
yali.wang.1@ulaval.ca

Marcus A. Brubaker
TTI Chicago
mbrubake@cs.toronto.edu

Brahim Chaib-draa
Laval University
chaib@ift.ulaval.ca

Raquel Urtasun
University of Toronto
urtasun@cs.toronto.edu

Abstract

In this paper we present a novel non-parametric approach to Bayesian filtering, where the prediction and observation models are learned in an online fashion. Our approach is able to handle multimodal distributions over both models by employing a mixture model representation with Gaussian Processes (GP) based components. To cope with the increasing complexity of the estimation process, we explore two computationally efficient GP variants, sparse online GP and local GP, which help to manage computation requirements for each mixture component. Our experiments demonstrate that our approach can track human motion much more accurately than existing approaches that learn the prediction and observation models offline and do not update these models with the incoming data stream.

1 INTRODUCTION

Many real world problems involve high dimensional data. In this paper we are interested in *modeling* and *tracking* human motion. In this setting, dimensionality reduction techniques are widely employed to avoid the curse of dimensionality.

Linear approaches such as principle component analysis (PCA) are very popular as they are simple to use. However, they often fail to capture complex dependencies due to their assumption of linearity. Non-linear dimensionality reduction techniques that attempt to preserve the local structure of the manifold (*e.g.*, Isomap [21, 8], LLE [19, 14]) can capture more complex dependencies, but often suffer when the manifold assumptions are violated, *e.g.*, in the presence of noise.

Probabilistic latent variable models have the advantage of being able to take the uncertainties into account when learning the latent representations. Perhaps the most suc-

cessful model in the context of modelling human motion is the Gaussian process latent variable model (GPLVM) [12], where the non-linear mapping between the latent space and the high dimensional space is modeled with a Gaussian process. This provides powerful prior models, which have been employed for character animation [28, 26, 15] and human body tracking [24, 16, 25].

In the context of tracking, one is interested in estimating the state of a dynamic system. The most commonly used technique for state estimation is Bayesian filtering, which recursively estimates the posterior probability of the state of the system. The two key components in the filter are the prediction model, which describes the temporal evolution of the process, as well as the observation model which links the state and the observations. A parametric form is typically employed for both models.

Ko and Fox [10] introduced the GP-BayesFilter, which defines the prediction and observation models in a non-parametric way via Gaussian processes. This approach is well suited when accurate parametric models are difficult to obtain. Its main limitation, however, resides in the fact that it requires ground truth states (as GPs are supervised), which are typically not available. GPLVMs were employed in [11] to learn the latent space in an unsupervised manner, bypassing the need for labeled data. This, however, can not exploit the incoming stream of data available in the online setting as the latent space is learned offline. Furthermore, only unimodal prediction and observation models can be captured due to the fact that the models learned by GP are nonlinear but Gaussian.

In this paper we extend the previous non-parametric filters to learn the latent space in an online fashion as well as to handle multimodal distributions for both the prediction and observation models. Towards this goal, we employ a mixture model representation in the particle filtering framework. For the mixture components, we investigate two computationally efficient GP variants which can update the prediction and observation models in an online fashion, and cope with the growth in complexity as the number of data points increases over time. More specifically, the sparse

online GP [3] selects the active set in an online fashion to efficiently maintain sparse approximations to the models. Alternatively, the local GP [26] reduces the computation by imposing local sparsity.

We demonstrate the effectiveness of our approach on a wide variety of motions, and show that both approaches perform better than existing algorithms. In the remainder of the paper we first present a review on Bayesian filtering and the GPLVM. We then introduce our algorithm and show our experimental evaluation followed by the conclusions.

2 BACKGROUND

In this section we review Bayesian filtering and Gaussian process latent variable models.

2.1 BAYESIAN FILTERING

Bayesian filtering is a sequential inference technique typically employed to perform state estimation in dynamic systems. Specifically, the goal is to recursively compute the posterior distribution of the current hidden state \mathbf{x}_t given the history of observations $\mathbf{y}_{1:t} = (\mathbf{y}_1, \dots, \mathbf{y}_t)$ up to the current time step

$$p(\mathbf{x}_t | \mathbf{y}_{1:t}) \propto p(\mathbf{y}_t | \mathbf{x}_t) \int p(\mathbf{x}_t | \mathbf{x}_{t-1}) p(\mathbf{x}_{t-1} | \mathbf{y}_{1:t-1}) d\mathbf{x}_{t-1}$$

where $p(\mathbf{x}_t | \mathbf{x}_{t-1})$ is the *prediction model* that represents the system dynamics, and $p(\mathbf{y}_t | \mathbf{x}_t)$ is the *observation model* that represents the likelihood of an observation \mathbf{y}_t given the state \mathbf{x}_t .

One of the most fundamental Bayesian filters is the Kalman filter, which is a maximum-a-posteriori estimator for linear and Gaussian models. Unfortunately, it is often not applicable in practice since most real dynamical systems are non-linear and/or non-Gaussian. Two popular extensions for non-linear systems are the extended Kalman filter (EKF) and the unscented Kalman filter (UKF) [9]. However, similar to the Kalman filter, the performance of EKF and UKF is poor when the models are multimodal [5].

In contrast, particle filters that are not restricted to linear and Gaussian models have been developed by using sequential Monte Carlo sampling to represent the underlying posterior $p(\mathbf{x}_t | \mathbf{y}_{1:t})$ [5]. More specifically, at each time step, N_p particles of \mathbf{x}_t are drawn from the prediction model $p(\mathbf{x}_t | \mathbf{x}_{t-1})$, and then all the particles are weighted according to the observation model $p(\mathbf{y}_t | \mathbf{x}_t)$. The posterior $p(\mathbf{x}_t | \mathbf{y}_{1:t})$ is approximated using these N_p weighted particles. Finally, the N_p particles are resampled for the next step. Unfortunately, the parametric description of the dynamic models limits the estimation accuracy of Bayesian filters.

Recently, a number of GP-based Bayesian filters were proposed by learning the prediction and observation models using GP regression [10, 4]. This is a promising alternative as GPs are non-parametric and can capture complex mappings. However, training these methods requires access to ground truth data before filtering. Unfortunately, the inputs of the training set are the hidden states which are not always known in real-world applications. Two extensions were introduced to learn the hidden states of the training set via a non-linear latent variable model [11] or a sparse pseudo-input GP regression [22]. However, these methods require offline learning procedures, which are not able to exploit the incoming data streams. In contrast, we propose two non-parametric particle filters that are able to exploit the incoming data to learn better models in an online fashion.

2.2 GAUSSIAN PROCESS DYNAMICAL MODEL

The Gaussian Process Latent Variable Model (GPLVM) is a probabilistic dimensionality reduction technique, which places a GP prior on the observation model [12]. Wang et al. [28] proposed the Gaussian Process Dynamical Model (GPDM), which enriches the GPLVM to capture temporal structure by incorporating a GP prior over the dynamics in the latent space. Formally, the model is:

$$\begin{aligned} \mathbf{x}_t &= f_{\mathbf{x}}(\mathbf{x}_{t-1}) + \eta_{\mathbf{x}} \\ \mathbf{y}_t &= f_{\mathbf{y}}(\mathbf{x}_t) + \eta_{\mathbf{y}} \end{aligned}$$

where $\mathbf{y} \in \mathbb{R}^{D_y}$ represents the observation and $\mathbf{x} \in \mathbb{R}^{D_x}$ the latent state, with $D_y \gg D_x$. The noise processes are assumed to be Gaussian $\eta_{\mathbf{x}} \sim \mathcal{N}(0, \sigma_{\mathbf{x}}^2 I)$ and $\eta_{\mathbf{y}} \sim \mathcal{N}(0, \sigma_{\mathbf{y}}^2 I)$. The nonlinear functions $f_{\mathbf{x}}^i$ and $f_{\mathbf{y}}^i$ have GP priors, i.e., $f_{\mathbf{x}}^i \sim \mathcal{GP}(0, k_{\mathbf{x}}(\mathbf{x}, \mathbf{x}'))$ and $f_{\mathbf{y}}^i \sim \mathcal{GP}(0, k_{\mathbf{y}}(\mathbf{x}, \mathbf{x}'))$ where $k_{\mathbf{x}}(\cdot, \cdot)$ and $k_{\mathbf{y}}(\cdot, \cdot)$ are the kernel functions. For simplicity, we denote the hyperparameters of the kernel functions by θ .

Let $\mathbf{x}_{1:T_0} = (\mathbf{x}_1, \dots, \mathbf{x}_{T_0})$ be the latent space coordinates from time $t = 1$ to time $t = T_0$. GPDM is typically learned by minimizing the negative log posterior $-\log(p(\mathbf{x}_{1:T_0}, \theta | \mathbf{y}_{1:T_0}))$ with respect to $\mathbf{x}_{1:T_0}$, and θ [28]. After $\mathbf{x}_{1:T_0}$ and θ are obtained, a standard GP prediction is used to construct the model $p(\mathbf{x}_t | \mathbf{x}_{t-1}, \theta, \mathcal{X}_{T_0})$ and $p(\mathbf{y}_t | \mathbf{x}_t, \theta, \mathcal{Y}_{T_0})$ with data $\mathcal{X}_{T_0} = \{(\mathbf{x}_{k-1}, \mathbf{x}_k)\}_{k=2}^{T_0}$ and $\mathcal{Y}_{T_0} = \{(\mathbf{x}_k, \mathbf{y}_k)\}_{k=1}^{T_0}$. Tracking ($t > T_0$) is then performed assuming the model is fixed and can be done using, e.g., a particle filter as described above. The major drawback of this approach is that it is not able to adapt to new observations during tracking. As shown in our experimental evaluation, this results in poor performance when the training set is small.

3 ONLINE GP PARTICLE FILTER

In order to solve the above-mentioned difficulties in learning and filtering with dynamic systems, we propose an *Online GP Particle Filter* framework to learn and refine the model during tracking, *i.e.*, the prediction $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ and observation $p(\mathbf{y}_t|\mathbf{x}_t)$ models are updated online in the particle filtering framework. To account for multi-modality and the significant amount of uncertainty that can be present, we propose to represent the prediction and observation models by a mixture model. For each mixture component, we will investigate two different GP variants.

Let the prediction and observation models at $t-1$ be

$$\begin{aligned} p(\mathbf{x}_t|\mathbf{x}_{t-1}, \Theta_{t-1,M}) &= \frac{1}{R_M} \sum_{i=1}^{R_M} p(\mathbf{x}_t|\mathbf{x}_{t-1}, \Theta_{t-1,M}^i) \\ p(\mathbf{y}_t|\mathbf{x}_t, \Theta_{t-1,O}) &= \frac{1}{R_O} \sum_{i=1}^{R_O} p(\mathbf{y}_t|\mathbf{x}_t, \Theta_{t-1,O}^i) \end{aligned} \quad (2)$$

where $\Theta_{t-1,M}^i$ and $\Theta_{t-1,O}^i$ represents the parameters of the i -th component, $\Theta_{t-1,M} = \{\Theta_{t-1,M}^i\}_{i=1}^{R_M}$ and $\Theta_{t-1,O} = \{\Theta_{t-1,O}^i\}_{i=1}^{R_O}$ are the parameters of all components. At the t -th time step, we run a standard particle filter to obtain a number of weighted particles. The latent space representations at time t can be obtained by resampling the weighted particles. Then, we assign each particle to the most likely mixture component of $p(\mathbf{x}_t|\mathbf{x}_{t-1}, \Theta_{t-1,M})$ and $p(\mathbf{y}_t|\mathbf{x}_t, \Theta_{t-1,O})$ to capture the multi-modality of the prediction and observation models. Finally, we compute the mean latent states of the assigned particles and use this mean state to update the corresponding components parameters, $\Theta_{t,M}^i$ for the prediction (or motion) model and $\Theta_{t,O}^i$ for the observation model. The whole framework is summarized in Algorithm 1.

What remains is to specify how the parameters of individual components are represented and updated (lines 18 and 23 in Algorithm 1). As noted above, we aim to use a GP model for each mixture component. However, a standard implementation would require $O(t^3)$ operations and $O(t^2)$ memory. As t grows linearly over time, the particle filter will quickly become too computationally and memory intensive. Thus a primary challenge is how to efficiently update the GP mixture components in the prediction and observation models.

In order to efficiently update $\Theta_{t,M}^i$ and $\Theta_{t,O}^i$ in an online manner, we consider two fast GP-based strategies: Sparse Online GP (SOGP) and Local GPs (LGP) in which the reduction in memory and/or computation is achieved by an online sparsification and a local experts mechanism respectively. A detailed review of fast GP approaches can be found in [1, 18].

The specific contents of $\Theta_{t,M}^i$ or $\Theta_{t,O}^i$ will vary depending on the method used. In the case of SOGP it will contain some computed quantities and the active set while for LGP it will simply be the set of all training points. While we will

Algorithm 1 Online GP-Particle Filter

```

1: Initialize model parameters  $\Theta$  based on  $\mathbf{y}_{1:T_0}$ 
2: Initialize particle set  $\mathbf{x}_{T_0}^{(1:N_P)}$  based on  $\mathbf{y}_{1:T_0}$ 
3: for  $t = T_0 + 1$  to  $T$  do
4:   for  $i = 1$  to  $N_p$  do
5:      $\mathbf{x}_t^{(i)} \sim p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, \Theta_{t-1,M})$ 
6:      $\hat{w}_t^{(i)} = p(\mathbf{y}_t|\mathbf{x}_t^{(i)}, \Theta_{t-1,O})$ 
7:   end for
8:   Normalize weights  $w_t^{(i)} = \hat{w}_t^{(i)} / (\sum_{i=1}^{N_p} \hat{w}_t^{(i)})$ 
9:   Resample particle set with probabilities  $w_t^{(1:N_p)}$ 
10:  for  $i = 1$  to  $N_p$  do
11:     $\eta_M^i = \arg \max_j p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)}, \Theta_{t-1,M}^j)$ 
12:     $\eta_O^i = \arg \max_j p(\mathbf{y}_t|\mathbf{x}_t^{(i)}, \Theta_{t-1,O}^j)$ 
13:  end for
14:  for  $j = 1$  to  $R_M$  do
15:     $n_{t-1}^j = \sum_{i=1}^{N_p} \delta(\eta_M^i = j)$ 
16:     $\bar{\mathbf{x}}_{t-1}^j = \frac{1}{n_{t-1}^j} \sum_{i=1}^{N_p} \delta(\eta_M^i = j) \mathbf{x}_{t-1}^{(i)}$ 
17:     $\bar{\mathbf{x}}_t^j = \frac{1}{n_{t-1}^j} \sum_{i=1}^{N_p} \delta(\eta_M^i = j) \mathbf{x}_t^{(i)}$ 
18:    Update  $\Theta_{t,M}^j$  with  $(\bar{\mathbf{x}}_{t-1}^j, \bar{\mathbf{x}}_t^j)$ 
19:  end for
20:  for  $j = 1$  to  $R_O$  do
21:     $n_{t-1}^j = \sum_{i=1}^{N_p} \delta(\eta_O^i = j)$ 
22:     $\bar{\mathbf{x}}_t^j = \frac{1}{n_{t-1}^j} \sum_{i=1}^{N_p} \delta(\eta_O^i = j) \mathbf{x}_t^{(i)}$ 
23:    Update  $\Theta_{t,O}^j$  with  $(\bar{\mathbf{x}}_t^j, \mathbf{y}_t)$ 
24:  end for
25: end for

```

focus on these two strategies, we note that in principle any similar update strategy could be used instead, such as informative vector machines [13] or local regression approaches [7, 6, 20]. In what follows, to avoid confusion with the notations of the latent state and observation, we will use \mathbf{a} and \mathbf{b} to indicate the input and output when we describe SOGP and LGP regression in which we consider modeling a generic function $\mathbf{b} = f(\mathbf{a}) + \xi$, with $\xi \sim \mathcal{N}(0, \sigma^2 I)$.

3.1 SPARSE ONLINE GAUSSIAN PROCESS

The Sparse Online Gaussian Process (SOGP) of [3, 27] is a well-known algorithm for online learning of GP models. To cope with the fact that data arrives in an online manner, SOGP trains a GP model sequentially by updating the posterior mean and covariance of the latent function values of the training set. This online procedure is coupled with a sparsification strategy which iteratively selects a fixed-size subset of points to form the active set, preventing the otherwise unbounded growth of the computation and memory load.

The key of SOGP is to maintain the joint posterior over the latent function values of the fixed-size active set \mathcal{D}_{t-1} , *i.e.*,

$\mathcal{N}(\mu_{t-1}, \Sigma_{t-1})$, by recursively updating μ_{t-1} and Σ_{t-1} . When a new observation $(\mathbf{a}_t, \mathbf{b}_t)$ ¹ is available, we perform the following update to take the new data point into account [27]

$$\mathbf{q}_t = Q_{t-1} \mathbf{k}_{t-1}(\mathbf{a}_t) \quad (3)$$

$$\rho_t^2 = k(\mathbf{a}_t, \mathbf{a}_t) - \mathbf{k}_{t-1}(\mathbf{a}_t)^T Q_{t-1} \mathbf{k}_{t-1}(\mathbf{a}_t) \quad (4)$$

$$\hat{\sigma}_t^2 = \sigma^2 + \rho_t^2 + \mathbf{q}_t^T \Sigma_{t-1} \mathbf{q}_t \quad (5)$$

$$\delta_t = \begin{bmatrix} \Sigma_{t-1} \mathbf{q}_t \\ \rho_t^2 + \mathbf{q}_t^T \Sigma_{t-1} \mathbf{q}_t \end{bmatrix} \quad (6)$$

$$\mu_t = \begin{bmatrix} \mu_{t-1} \\ \mathbf{q}_t^T \mu_{t-1} \end{bmatrix} + \hat{\sigma}_t^{-2} (\mathbf{b}_t - \mathbf{q}_t^T \mu_{t-1}) \delta_t \quad (7)$$

$$\Sigma_t = \begin{bmatrix} \Sigma_{t-1} & \Sigma_{t-1} \mathbf{q}_t \\ \mathbf{q}_t^T \Sigma_{t-1} & \rho_t^2 + \mathbf{q}_t^T \Sigma_{t-1} \mathbf{q}_t \end{bmatrix} - \hat{\sigma}_t^{-2} \delta_t \delta_t^T \quad (8)$$

where $\mathbf{k}_{t-1}(\mathbf{a}_t)$ is the kernel vector which is constructed from \mathbf{a}_t and the active set \mathcal{D}_{t-1} , and Q_{t-1} is the inverse kernel matrix of the active set \mathcal{D}_{t-1} .

One of the key steps in this algorithm is how to decide when to add the new point to the active set. We employed the strategy suggested by [3, 27], and ignore the new point with $\rho_t^2 < \epsilon$ for some small value of ϵ (we use $\epsilon = 10^{-6}$). In this case, the μ_t, Σ_t are updated as $\mu_t \leftarrow [\mu_t]_{-i}, \Sigma_t \leftarrow [\Sigma_t]_{-i, -i}$ where $i = t$ is the index of the new point, $[\cdot]_{-i}$ removes the i -th entry of a vector, and $[\cdot]_{-i, -i}$ removes the i -th row and column of a matrix. Additionally, the inverse kernel matrix is simply $Q_t = Q_{t-1}$ because the new point is not included in the active set. When $\rho_t^2 \geq \epsilon$, we add the new point to the active set $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{(\mathbf{a}_t, \mathbf{b}_t)\}$. The μ_t, Σ_t are then the same as Eq.(7) and (8), and the inverse kernel matrix is updated to be [27]

$$Q_t = \begin{bmatrix} Q_{t-1} & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix} + \rho_t^{-2} \begin{bmatrix} \mathbf{q}_t \mathbf{q}_t^T & -\mathbf{q}_t \\ -\mathbf{q}_t^T & 1 \end{bmatrix} \quad (9)$$

When the size of the active set is larger than the fixed size N_A because a new point was added, we must remove a point. This is done by selecting the one which minimally affects the predictions according to the squared predicted error. Following [3, 27], we remove the j -th data point with

$$j = \arg \min_j \left(\frac{[Q_t \mu_t]_j}{[Q_t]_{j,j}} \right)^2 \quad (10)$$

where $[\cdot]_j$ selects the j -th entry of a vector and $[\cdot]_{j,j}$ select the j th diagonal entry of a matrix. Once a point has been selected for removal, μ_t, Σ_t and Q_t are updated as

$$\mu_t \leftarrow [\mu_t]_{-j} \quad (11)$$

$$\Sigma_t \leftarrow [\Sigma_t]_{-j, -j} \quad (12)$$

$$Q_t \leftarrow [Q_t]_{-j, -j} - \frac{[Q_t]_{-j,j} [Q_t]_{j,j}^T}{[Q_t]_{j,j}} \quad (13)$$

¹For simplicity of presentation, we assume that \mathbf{b} is a scalar. The extension to vector valued \mathbf{b} is straightforward.

Algorithm 2 SOGP Update

input Previous posterior quantities $\mu_{t-1}, \Sigma_{t-1}, Q_{t-1}$

input Previous active set \mathcal{D}_{t-1}

input New input-output observation $(\mathbf{a}_t, \mathbf{b}_t)$ pair

1: Compute ρ_t, μ_t and Σ_t as in Equations (4), (7) and (8).

2: **if** $\rho_t^2 < \epsilon$ **then**

3: Perform update $\mu_t \leftarrow [\mu_t]_{-i}, \Sigma_t \leftarrow [\Sigma_t]_{-i, -i}$ where i is the index of the newly added row to μ_t .

4: Set $Q_t = Q_{t-1}, \mathcal{D}_t = \mathcal{D}_{t-1}$.

5: **else**

6: Compute Q_t as in Equation (9).

7: Add to active set $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{(\mathbf{a}_t, \mathbf{b}_t)\}$.

8: **end if**

9: **if** $|\mathcal{D}_t| > N_A$ **then**

10: Select point j to remove using Equation (10).

11: Perform update $\mu_t \leftarrow [\mu_t]_{-j}, \Sigma_t \leftarrow [\Sigma_t]_{-j, -j}$ and

$$Q_t \leftarrow [Q_t]_{-j, -j} - \frac{[Q_t]_{-j,j} [Q_t]_{j,j}^T}{[Q_t]_{j,j}}.$$

12: Remove j from active set $\mathcal{D}_t \leftarrow \mathcal{D}_t \setminus \{(\mathbf{a}_j, \mathbf{b}_j)\}$.

13: **end if**

output μ_t, Σ_t, Q_t and \mathcal{D}_t

where $[\cdot]_{-j,j}$ selects the j -th column of the matrix with the j -th row removed and the point is removed from the active set $\mathcal{D}_t \leftarrow \mathcal{D}_t \setminus \{(\mathbf{a}_j, \mathbf{b}_j)\}$.

The joint posterior at time t can be used to construct the predictive distribution for a new input \mathbf{a}^*

$$p^{SOGP}(\mathbf{b}|\mathbf{a}^*, \mathcal{D}_t, \Theta) = \mathcal{N}(\mathbf{b}|\tilde{\mathbf{b}}, \tilde{\sigma}^2) \quad (14)$$

where $\tilde{\mathbf{b}} = \mathbf{k}_t(\mathbf{a}^*)^T Q_t \mu_t$ and $\tilde{\sigma}^2 = \sigma^2 + k(\mathbf{a}^*, \mathbf{a}^*) + \mathbf{k}_t(\mathbf{a}^*)^T (Q_t \Sigma_t Q_t - Q_t) \mathbf{k}_t(\mathbf{a}^*)$. We summarize the SOGP updates in Algorithm 2.

3.2 LOCAL GAUSSIAN PROCESSES

An alternative to the SOGP approach is to use Local Gaussian Processes (LGP), which was developed specifically to deal with large, multi-modal regression problems [17, 23]. In LGP, given a test input \mathbf{a}^* and a set of input-output pairs $\mathcal{D} = \{(\mathbf{a}_i, \mathbf{b}_i)\}_{i=1}^N$, the $M_{\mathbf{a}}$ -nearest neighbors $\mathcal{D}_{\mathbf{a}^*} = \{(\mathbf{a}_\ell, \mathbf{b}_\ell)\}_{\ell=1}^{M_{\mathbf{a}}}$ are selected based on the distance in the input space $d_\ell = \|\mathbf{a}_\ell - \mathbf{a}^*\|$. Then, for each of the $M_{\mathbf{a}}$ neighbors, $M_{\mathbf{b}}$ -nearest neighbors $\mathcal{D}_{\mathbf{b}_\ell} = \{(\mathbf{a}_j, \mathbf{b}_j)\}_{j=1}^{M_{\mathbf{b}}}$ are selected based on the distance in the output space to \mathbf{b}_ℓ . These neighbors are then combined to form a local GP expert which makes a Gaussian prediction with mean and covariance

$$\mu_\ell = B_{\mathcal{D}_{\mathbf{b}_\ell}} K_{\mathcal{D}_{\mathbf{b}_\ell}, \mathcal{D}_{\mathbf{b}_\ell}}^{-1} \mathbf{k}_{\mathcal{D}_{\mathbf{b}_\ell}}(\mathbf{a}^*)$$

$$\sigma_\ell^2 = k(\mathbf{a}^*, \mathbf{a}^*) - \mathbf{k}_{\mathcal{D}_{\mathbf{b}_\ell}}(\mathbf{a}^*)^T K_{\mathcal{D}_{\mathbf{b}_\ell}, \mathcal{D}_{\mathbf{b}_\ell}}^{-1} \mathbf{k}_{\mathcal{D}_{\mathbf{b}_\ell}}(\mathbf{a}^*) + \sigma^2$$

where $B_{\mathcal{D}_{\mathbf{b}_\ell}}$ is the matrix whose columns are the $M_{\mathbf{b}}$ nearest neighbors of \mathbf{b}_ℓ , $\mathbf{k}_{\mathcal{D}_{\mathbf{b}_\ell}}(\mathbf{a}^*)$ is the vector of kernel function values for the input \mathbf{a}^* and the points in $\mathcal{D}_{\mathbf{b}_\ell}$, and

$K_{\mathcal{D}_{\mathbf{b}_\ell}, \mathcal{D}_{\mathbf{b}_\ell}}$ is the kernel matrix for the points in $\mathcal{D}_{\mathbf{b}_\ell}$. The final predictive distribution is then formed by combining all local experts in a mixture model

$$p^{LGP}(\mathbf{b}|\mathbf{a}^*, \mathcal{D}, \Theta) = \sum_{\ell=1}^{M_a} w_\ell \mathcal{N}(\mathbf{b}|\mu_\ell, \sigma_\ell^2 I) \quad (15)$$

with weights $w_\ell \propto 1/d_\ell$.

4 EXPERIMENTAL EVALUATION

To illustrate our approach we choose 4 very different motions, *i.e.*, walking, golf swing, swimming as well as exercises (composed of side twist and squat). The data consists of motion capture from the CMU dataset [2], where each observation is a 62 dimensional vector containing the 3D rotations of all joints. We normalize the data to be mean zero and subsample the observations to reduce the correlation between consecutive frames. We use a frequency of 12 frames/s for walking and swimming, 24 frames/s for golf swing and 30 frames/s for the exercise motion. We compute all results averaged over 3 trials and report the average root mean squared error as our measure of performance.

In all the experiments, the latent space dimensionality is set to be 3 as is common for human motion models [28]. We use PCA to initialize the latent space and K-means to obtain the data points used for the mixture components. We choose the compound kernel function $k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp(-0.5 \|\mathbf{x} - \mathbf{x}'\|^2 / \gamma^2) + l^2 \mathbf{x}^T \mathbf{x}'$ for both prediction and observation mappings. Unless otherwise stated, we use 50 particles, a training set of size of 20/30/50/450 and 2/2/5/5 mixture components for walking/golf/swimming/exercise motions respectively. For LGP, the number of local GP experts is 2/2/2/5, and the size of each local expert is 5/8/5/20. For SOGP, the size of the active set is 20/5/50/20. The parameter values were chosen to balance computational cost with the prediction accuracy and in our experiments we demonstrate the robustness of our approach to these parameters.

4.1 COMPARISON TO STATE-OF-THE-ART

We compare our approaches to two baselines: The first one is the approach of Ko and Fox [11] where a GPDM is learned offline with gradient descent [28] before performing particle filtering for state estimation. The second baseline is similar, but learns the GPDM offline using stochastic gradient descent [29]. We tested the baselines in two different settings. First, only the initial training set is available to learn the prediction and observation models. Second, all the data (including future streamed examples) are used to learn the prediction and observation models. Note that the latter represents the oracle for Ko and Fox [11].

Number of Particles: We evaluate how the accuracy changes as a function of the number of particles, N_p . As

expected, the prediction error is reduced in all the methods when the number of particles increases. As shown in the first row of Fig. 1, our approaches are superior to the baselines. Importantly, we outperformed the oracle baseline as we are able to represent multi-modal distributions effectively. This is particularly important in the exercise sequence as the dynamics are clearly multimodal due to the different motions that are performed in the sequence. Furthermore, our LGP variant outperforms SOGP. We believe this is due to the fact that SOGP has a fixed capacity while LGP is able to leverage more training data when making predictions.

Influence of noise: In this experiment we evaluate the robustness of all approaches to additive noise in the observations. The second row of Fig. 1 shows that our LGP particle filter significantly outperforms the baselines, particularly in the exercise sequence. Our SOGP outperforms all baselines that have access to the same training set, and is only beaten by the oracle for walking.

Size of Training Set: We next evaluate how the accuracy depends on the size of the initial training set, T_0 . The first row of Fig. 2 clearly indicates that our methods perform well even when the training set is very small. In contrast, the two baselines require bigger training sets to achieve comparable performance. This is expected as the baselines do not update the latent space to take the incoming observations into account.

4.2 QUALITATIVE EXPERIMENTS

Fig. 3 shows the latent space of both SOGP and LGP filters when employing 50 particles for each time step (depicted in blue). From the 3D latent space and predicted skeletons, we find that the manifolds of both LGP and SOGP particle filters have a good representation of the high-dimensional human motion data.

4.3 PROPERTIES OF OUR METHODS

We next discuss various aspects of our method and evaluate the influence of the parameters of SOGP and LGP filters. For LGP, due to the fact that the data sizes of walking, golf and swimming motions are small, we reduced the number (size) of local experts to be able to increase the size (number) of the local experts.

Computational Complexity: Overall the computational complexity of our method (Alg 1) is mainly determined by the complexity of constructing a prediction distribution for each components (lines 5-6 and 11-12) and model updates (line 18 and line 23). Specifically, for an individual component which is either SOGP or LGP, computing the prediction distribution is $O(N_A^2)$ or $O(M_a M_b^3 + T M_a M_b)$ respectively where N_A is the size of active set, M_a is the number of local experts, M_b are the number of neigh-

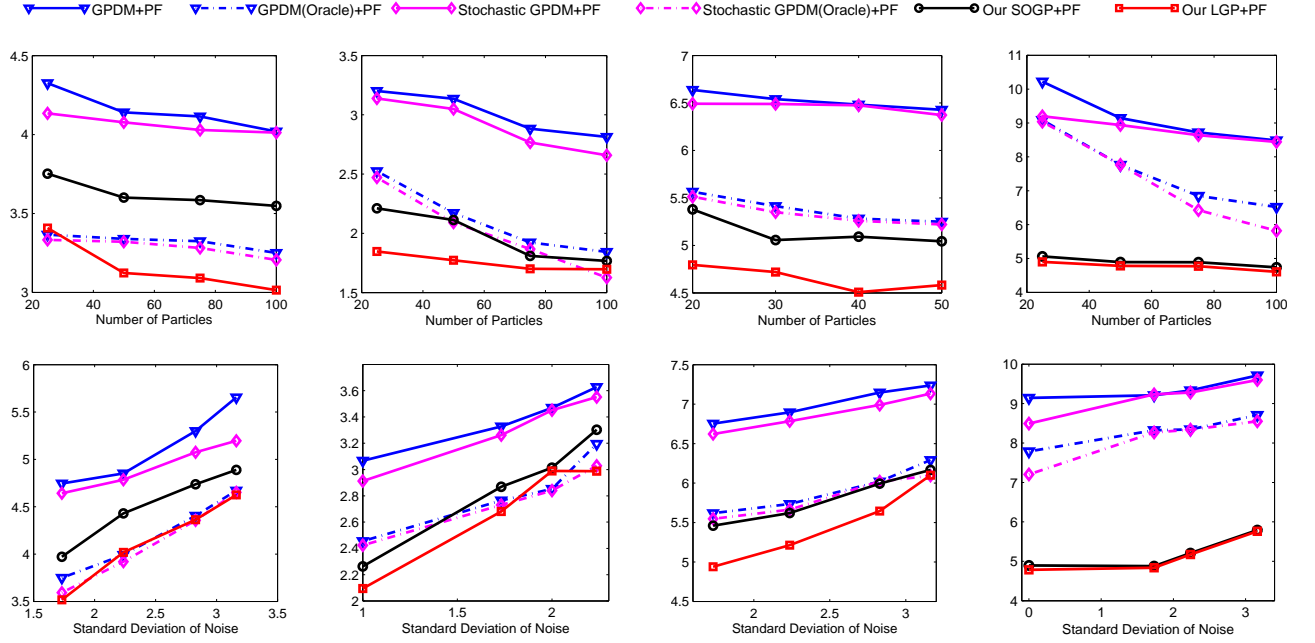


Figure 1: Root mean squared error as a function of (1st row) the number of particles in the particle filter, (2nd row) the standard deviation of noise added to the observations. The columns (from left to right) represent walking, golf, swimming and exercise motions. Note that our approach outperforms the baselines in all settings. Handling multimodal distributions is particularly important in the exercise example as it is composed of a variety of different motions.

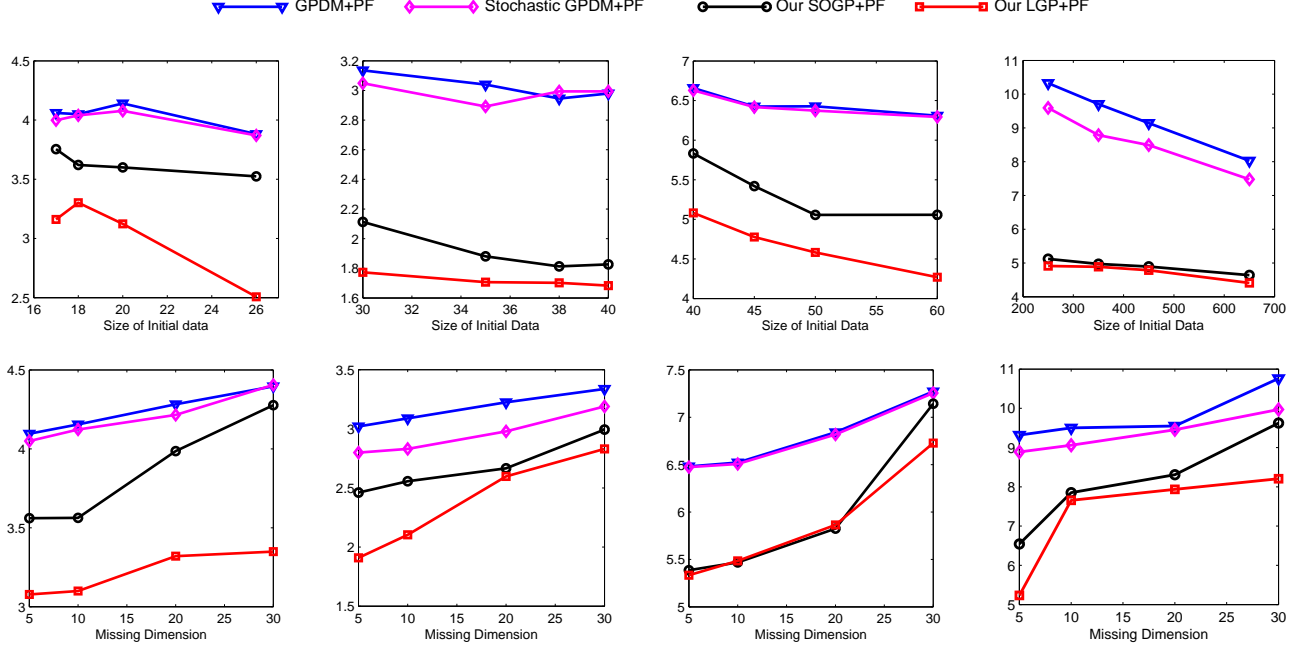


Figure 2: Root Mean Squared Error as a function of (1st row) the number of initial training points, (2nd row) the number of the missing dimensions. The columns (from left to right) are respectively for walking, golf, swimming and exercise motions.

bors in the output space and $TM_a M_b$ comes from the KNN search. The model updates for the mixture compo-

nents (lines 18 and 23) have a computational complexity of $O(N_A^2)$ and $O(1)$ for SOGP and LGP respectively.

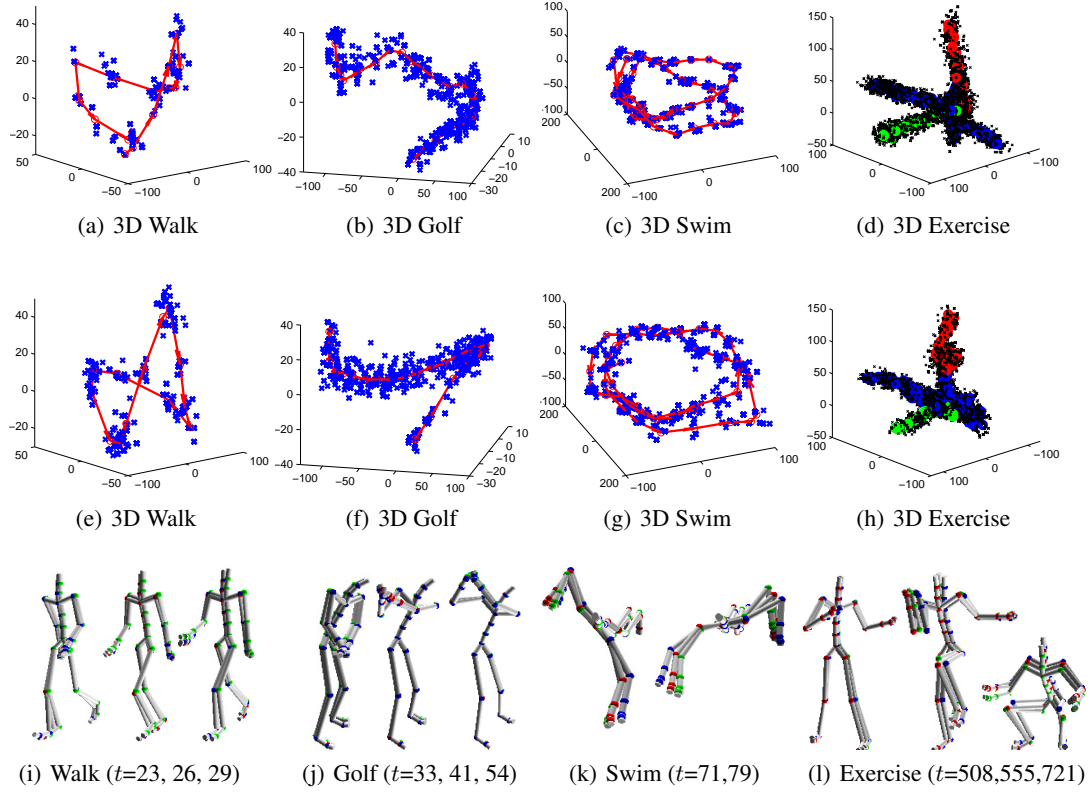


Figure 3: 3D latent spaces learned while tracking: The first row depicts the results of our SOGP variant, while the second row shows our LGP variant. In the walk/golf/swimming plots the red curve represents the predicted mean of the latent state sequence and the blue crosses are the particles at each step. In the plots for exercise (last column), the red, blue and green curves are the predicted mean of the latent state for three motions in the exercise sequence, and the black crosses are the particles at each step. The third row depicts the predicted skeletons, where ground truth is shown in green, our SOGP variant in blue and our LGP variant in red.

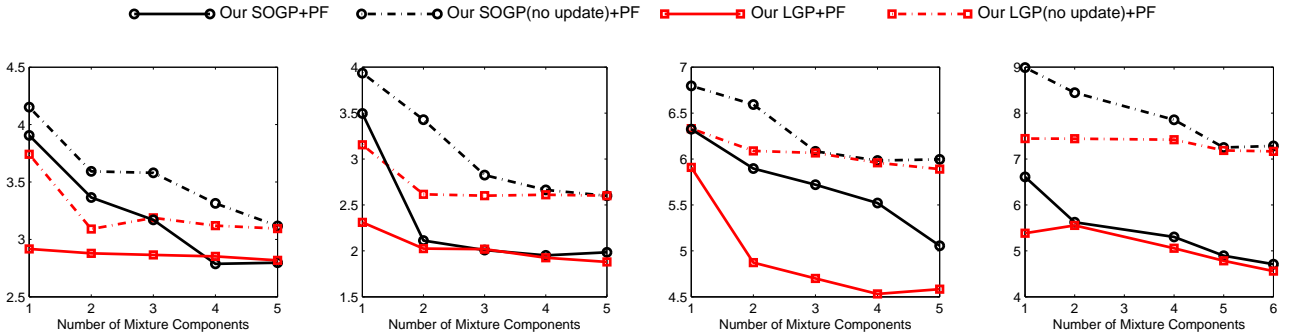


Figure 4: Root Mean Squared Error as a function of the number of mixture components. The columns (from left to right) are respectively for walking, golf, swimming and exercise motions.

Number of Mixture Components: Fig. 4 shows performance as a function of the number of mixture components, R_M and R_O , for both SOGP and LGP. For LGP+PF in walk/golf/swim/exercise, the number of local GPs are 1/1/2/5 and the size of each local GP are 3/3/5/20. In all cases, we set $R_M = R_O$. Note that performance typi-

cally increases with the number of mixture components, for SOGP, but less so for LGP. Furthermore, our approaches outperform the baselines in which the model is not updated during filtering indicating that the online model updating is very important in practice. Also note that while LGP generally outperforms SOGP, the difference quickly declines

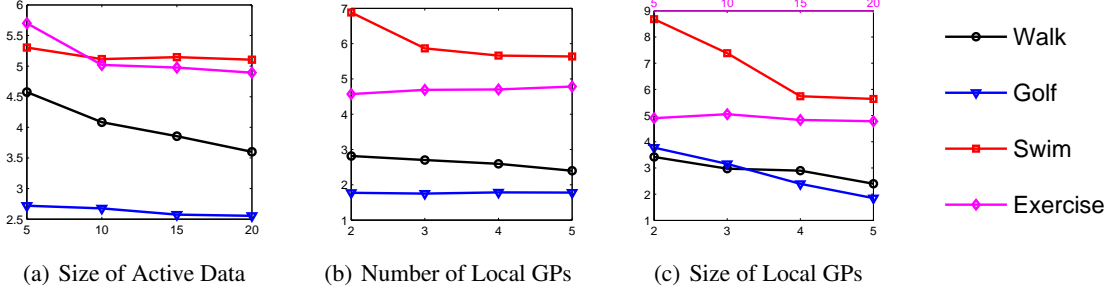


Figure 5: Root mean squared error as a function of the size of the active set in SOGP, the number of the local GP experts and the size of each local GP expert in LGP. In the subplot 5(c), the top x -axis is for exercise and the bottom one for the other motions.

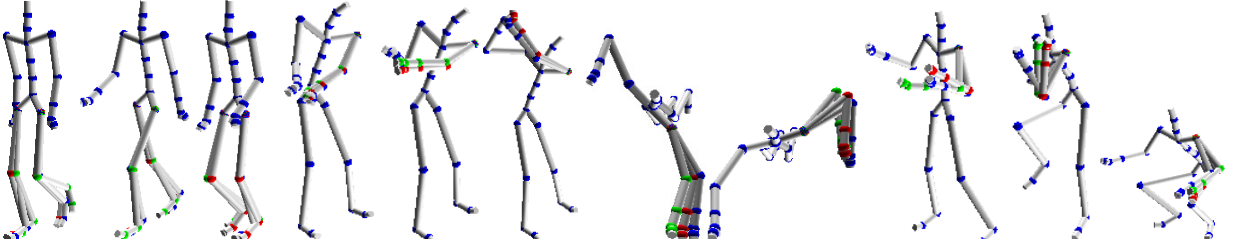


Figure 6: Predicted skeleton for missing parts (Walk: two legs; Golf, Swim and Exercise: left arm). The ground truth is shown in green, our SOGP particle filter in blue and LGP particle filter in red. We show the predicted performance at $t=24, 27, 32$ for walk, $t=34, 38, 50$ for golf, $t=71, 79$ for swim, $t=470, 592, 711$ for exercise.

as the number of mixture components increases. This suggests that, when the fixed memory requirements of SOGP is desirable, a larger number of mixture components will achieve performance comparable to LGP.

Active Set size in SOGP: To explore the effect of the size of the active set, N_A , on performance we set the number of mixture components, R_M and R_O , to be 2/1/5/5 for walk/golf/swim/exercise, and use the same settings as before for the other parameters. Results are shown in Fig. 5(a). As expected performance improves when the size of the active set increases.

Number and Size of Local Experts in LGP: Figs. 5(b) and 5(c) show the performance of our approach as a function of the number of local GP experts, M_a , as well as their size, M_b . For this experiment we set the number of mixture components, R_M and R_O , to be 1/2/1/5 and used the same settings as before for the other parameters except when evaluating the size of each local GP expert where we set the number of local GP experts to 5/2/5/5. As shown in the figure, even with the small number (size) of local GP experts, we still achieve good performance.

4.4 HANDLING MISSING DATA

In this setting, we evaluate the capabilities of our approaches to handle missing data. We assume that the initial set has no missing values, but a fixed set of joint angles are

missing for all incoming frames. Our approach is able to cope with missing data with only two small modifications. First, particles are weighted only based on the observed dimensions. Furthermore, when updating the prediction and observation models, we employ mean imputation for the missing observation dimensions. Fig. 6 shows reconstructions of the missing dimensions for all our motions, which consists of the two legs for walking, the left arm for golf swing, swimming and exercise motions. We can see that our approach is able to reconstruct the missing parts well.

Finally, to evaluate the tracking performance as a function of the number of missing dimensions, we randomly generate the indices for the missing dimensions and use the same missing dimensions for all incoming frames. The second row of Fig. 2 shows that, compared to the baselines, our methods perform well even when the number of missing dimensions is 20 (1/3 of the skeleton) for all the motions. In addition, our LGP particle filter outperforms our SOGP variant.

5 CONCLUSION

In this paper we have presented a novel non-parametric approach to Bayesian filtering, where the observation and prediction models are constructed using a mixture model with GP components learned in an online fashion. We have demonstrated that our approach can capture the mul-

timodality accurately and efficiently by online updates. We have explored two fast GP variants for updating which keep memory and computation bounded for individual mixture components. We have demonstrated the effectiveness of our approach when tracking different human motions and explored the impact of various parameters on performance. The Local GP particle filter proved superior to our SOGP variant, however these differences can be mitigated by using more mixture components when using SOGP. In the future, we plan to investigate the usefulness of our approach in other settings such as shape deformation estimation and financial time series.

References

- [1] K. Chalupka, C. K. I. Williams, and I. Murray. A Framework for Evaluating Approximation Methods for Gaussian Process Regression. *JMLR*, 2013.
- [2] CMU Mocap Database. CMU Mocap Database. <http://mocap.cs.cmu.edu/>.
- [3] L. Csato and M. Opper. Sparse online gaussian processes. In *Neural Computation*, 2002.
- [4] M. P. Deisenroth, M. F. Huber, and U. D. Hanebeck. Analytic moment-based gaussian process filtering. In *ICML*, 2009.
- [5] A. Doucet, S. Godsill, and C. Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and Computing*, 2000.
- [6] T. Hastie and C. Loader. Local Regression: Automatic Kernel Carpentry. *Statistical Science*, 1993.
- [7] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive Mixtures of Local Experts. *Neural Computation*, 1991.
- [8] O. C. Jenkins and M. Matarić. A spatio-temporal extension to Isomap nonlinear dimension reduction. In *ICML*, 2004.
- [9] S. J. Julier, J. K. Uhlmann, and H. F. Durrant-Whyte. A new approach for filtering nonlinear systems. In *American Control Conference*, 1995.
- [10] J. Ko and D. Fox. Gp-bayesfilters: Bayesian filtering using gaussian process prediction and observation models. In *IROS*, 2008.
- [11] J. Ko and D. Fox. Learning gp-bayesfilters via gaussian process latent variable models. In *RSS*, 2009.
- [12] N. Lawrence. Probabilistic non-linear principal component analysis with gaussian process latent variable models. *JMLR*, 6:1783–1816, 2005.
- [13] N. Lawrence, M. Seeger, and R. Herbrich. Fast Sparse Gaussian Process Methods: The Informative Vector Machine. In *NIPS*, 2003.
- [14] C-S. Lee and A. Elgammal. Coupled Visual and Kinematics Manifold Models for Human Motion Analysis. *IJCV*, 2010.
- [15] S. Levine, J. Wang, A. Haraux, Z. Popovic, and V. Koltun. Continuous character control with low-dimensional embeddings. In *SIGGRAPH*, 2012.
- [16] K. Moon and V. Pavlovic. Impact of dynamics on subspace embedding and tracking of sequences. In *CVPR*, pages 198–205, 2006.
- [17] D. Nguyen-Tuong, J. Peters, and M. Seeger. Local gaussian process regression for real time online model learning and control. In *NIPS*, 2008.
- [18] J. Quiñero-Candela and C. E. Rasmussen. A Unifying View of Sparse Approximate Gaussian Process Regression. *JMLR*, 2005.
- [19] S. Roweis and L. Saul. Nonlinear Dimensionality Reduction by Locally Linear Embedding. *Science*, 290 (5500):2323–2326, 2000.
- [20] S. Schaal and C. G. Atkeson. Constructive Incremental Learning From Only Local Information. *Neural Computation*, 1998.
- [21] J. Tenenbaum, V. de Silva, and J. Langford. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science*, 2000.
- [22] R. Turner, M. P. Deisenroth, and C. E. Rasmussen. State-space inference and learning with gaussian processes. In *AISTATS*, 2010.
- [23] R. Urtasun and T. Darrell. Sparse probabilistic regression for activity-independent human pose inference. In *CVPR*, 2008.
- [24] R. Urtasun, D. Fleet, A. Hertzman, and P. Fua. Priors for people tracking from small training sets. In *ICCV*, 2005.
- [25] R. Urtasun, D. Fleet, and P. Fua. 3d people tracking with gaussian process dynamical models. In *CVPR*, 2006.
- [26] R. Urtasun, D. Fleet, A. Geiger, J. Popovic, T. Darrell, and N. Lawrence. Topologically-constrained latent variable models. In *ICML*, 2008.
- [27] S. Van Vaerenbergh, M. Lázaro-Gredilla, and I. Santamaría. Kernel recursive least-squares tracker for time-varying regression. *IEEE Transactions on Neural Networks and Learning Systems*, 2012.
- [28] J. Wang, D. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *PAMI*, 30 (2):283–298, 2008. ISSN 0162-8828.
- [29] A. Yao, J. Gall, L. V. Gool, and R. Urtasun. Learning probabilistic non-linear latent variable models for tracking complex activities. In *NIPS*, 2011.

Approximating the Bethe Partition Function

Adrian Weller

Department of Computer Science
Columbia University
New York NY 10027
adrian@cs.columbia.edu

Tony Jebara

Department of Computer Science
Columbia University
New York NY 10027
jebara@cs.columbia.edu

Abstract

When belief propagation (BP) converges, it does so to a stationary point of the Bethe free energy \mathcal{F} , and is often strikingly accurate. However, it may converge only to a local optimum or may not converge at all. An algorithm was recently introduced by Weller and Jebara for attractive binary pairwise MRFs which is guaranteed to return an ϵ -approximation to the global minimum of \mathcal{F} in polynomial time provided the maximum degree $\Delta = O(\log n)$, where n is the number of variables. Here we extend their approach and derive a new method based on analyzing first derivatives of \mathcal{F} , which leads to much better performance and, for attractive models, yields a fully polynomial-time approximation scheme (FPTAS) without any degree restriction. Further, our methods apply to general (non-attractive) models, though with no polynomial time guarantee in this case, demonstrating that approximating \log of the Bethe partition function, $\log Z_B = -\min \mathcal{F}$, for a general model to additive ϵ -accuracy may be reduced to a discrete MAP inference problem. This allows the merits of the global Bethe optimum to be tested.

1 INTRODUCTION

Undirected graphical models, also termed Markov random fields (MRFs), are central tools in machine learning. A set of variables and a score function is specified such that the probability of a configuration of variables is proportional to the value of the score function, which factorizes into sub-functions over subsets of variables in a way that defines a topology on the variables.

Three central problems are: (1) To evaluate the partition function Z , which is the sum of the score function over all possible settings, and hence is the normalization con-

stant for the probability distribution; (2) Marginal inference, which is computing the probability distribution of a given subset of variables; and (3) Maximum a posteriori (MAP) inference, which is the task of identifying a setting of all the variables which has maximum probability.

All these are NP-hard, and (1) and (2) are closely related (marginals are a ratio of two partition functions). Variational methods show that the partition function may be obtained by minimizing the free energy over the marginal polytope, and that if instead the Bethe free energy (Bethe, 1935) is minimized over the local polytope, this should yield a good approximation¹. Although this is not a formal result, and there are cases where it performs poorly - typically when there are many short cycles with strong edge interactions (Wainwright and Jordan, 2008, § 4.1), still, the approach has proved very popular and often strikingly accurate. Belief propagation is often used to perform this minimization (Pearl, 1988; Yedidia et al., 2001). Performance is often excellent (McEliece et al., 1998; Murphy et al., 1999), but when applied to models with cycles, termed loopy belief propagation (LBP), convergence is not guaranteed in general, even to a local minimum. Some conjectured that when LBP behaves poorly, it is likely that the Bethe approximation, as given by the global minimum, also performs poorly, but it has not previously been possible to test this.

Approaches such as gradient descent (Welling and Teh, 2001), double-loop methods (Yuille, 2002) or Frank-Wolfe (Belanger et al., 2013) will converge but only to a local minimum, and with no runtime guarantee. Recently, two methods with polynomial runtime were given for the important subclass of binary pairwise models: one returns an approximately stationary point (Shin, 2012), though its value may be far even from a local minimum; the other returns an ϵ -approximate global optimum value (Weller and Jebara, 2013a) but only for the restricted case of attractive models (where pairwise relationships tend to pull connected variables to the same value)¹. Both these methods restrict the topology to have maximum degree $O(\log n)$,

¹All terms are defined in §2.

where n is the number of variables.

1.1 CONTRIBUTION AND SUMMARY

We obtain results for binary pairwise MRFs by expanding on ideas from Weller and Jebara (2013a). The approach is to construct a *sufficient mesh* of discretized points in such a way that the optimum mesh point q^* is guaranteed to have $\mathcal{F}(q^*)$ within ϵ of the true optimum. Our first derivative method typically results in a mesh that is much coarser (by many orders of magnitude, see §6.1), yet still sufficient, and admits adaptive methods to focus points in regions where \mathcal{F} may vary rapidly. This leads to a FPTAS for attractive models with no restriction on topology. In addition, we refine and extend the second derivative approach of Weller and Jebara (2013a) to derive a method that performs well for very small ϵ . With our new methods, both approaches apply to general binary pairwise models (not necessarily attractive) to reduce the problem of finding an ϵ -approximate global optimum to solving a derived discrete optimization problem, which may be framed as multi-label MAP inference, where a rich family of methods already exists.

There are several motivations for this work:

- To our knowledge, we present the first way to solve for the global Bethe optimum (within ϵ accuracy) of a general binary pairwise MRF. Runtime is practical for small real-world problems.
- This now allows the accuracy of the global Bethe optimum to be tested.
- For attractive models, we obtain a fully polynomial time approximation scheme for any topology, thus answering an open theoretical question.

In §2, we establish notation and present preliminary results, then apply these in §3 to derive our new approach for mesh construction based on analyzing first derivatives of \mathcal{F} . In §4 we revisit the second derivative approach of Weller and Jebara (2013a). We show how this method can be refined and extended to yield better performance and also to admit non-attractive models, though for most cases of interest, unless ϵ is very small, the method of §3 is much superior.

In §5, we discuss the resulting discrete optimization problem. In certain settings this is tractable, and in general we mention several features that can make it easier to find a satisfactory solution, or at least to bound its value. Experiments are described in §6 demonstrating practical application of the algorithm. Conclusions are presented in §7.

For a sketch of the overall approach, see Algorithm, 1.

1.2 RELATED WORK

Jerrum and Sinclair (1993) derived a fully polynomial-time randomized approximation scheme (FPRAS) for the true

Algorithm 1 Mesh method to return ϵ -approximate global optimum $\log Z_B$ for a general binary pairwise model

Input: ϵ , model parameters (convert using §2.1 if required)

Output: estimate of global optimum $\log Z_B$ guaranteed to be in range $[\log Z_B - \epsilon, \log Z_B]$, together with corresponding pseudo-marginal as arg for the discrete optimum

- 1: Preprocess by computing bounds on the locations of minima, see §2.4.
- 2: Construct a sufficient mesh using one of the methods in this paper, see §3 & 4. All approaches are fast, so several may be used and the most efficient mesh selected.
- 3: Attempt to solve the resulting multi-label MAP inference problem, see §5.
- 4: If unsuccessful, but a strongly persistent partial solution was obtained, then improved location bounds may be generated (see §5.2.1), repeat from 2.

At anytime, one may stop and compute bounds on $\log Z_B$, see §5.2.

partition function, but only when singleton potentials are uniform (i.e. a uniform external field), and the runtime is high at $O(\epsilon^{-2} m^3 n^{11} \log n)$. Heinemann and Globerson (2011) have shown that models exist such that the true marginal probability cannot possibly be the location of a minimum of the Bethe free energy. Approaches have been developed to solve related convex problems but results are typically less good (Meshi et al., 2009). Our work demonstrates an interesting connection between MAP inference techniques (NP-hard) and estimating the partition function Z (#P-hard). A different connection was shown by using MAP inference on randomly perturbed models to approximate and bound Z (Hazan and Jaakkola, 2012).

2 NOTATION & PRELIMINARIES

Our notation is similar to Weller and Jebara (2013a) and Welling and Teh (2001). We focus on a binary pairwise model with n variables $X_1, \dots, X_n \in \mathbb{B} = \{0, 1\}$ and graph topology $(\mathcal{V}, \mathcal{E})$ with $m = |\mathcal{E}|$; that is \mathcal{V} contains nodes $\{1, \dots, n\}$ where i corresponds to X_i , and $\mathcal{E} \subseteq V \times V$ contains an edge for each pairwise score relationship. Let $\mathcal{N}(i)$ be the neighbors of i . Let $x = (x_1, \dots, x_n)$ be one particular configuration, and introduce the notion of energy $E(x)$ through²

$$p(x) = \frac{e^{-E(x)}}{Z}, \quad E = - \sum_{i \in \mathcal{V}} \theta_i x_i - \sum_{(i,j) \in \mathcal{E}} W_{ij} x_i x_j, \quad (1)$$

²The probability or score function can always be reparameterized in this way, with finite θ_i and W_{ij} terms provided $p(x) > 0 \forall x$, which is a requirement for our approach. There are reasonable distributions where this does not hold, i.e. distributions where $\exists x : p(x) = 0$, but this can often be handled by assigning such configurations a sufficiently small positive probability ϵ .

where the partition function $Z = \sum_x e^{-E(x)}$ is the normalizing constant, and $\{\theta_i, W_{ij}\}$ are parameters of the model.

Given any joint probability distribution $p(X_1, \dots, X_n)$ over all variables, the (Gibbs) free energy is defined as $\mathcal{F}_G(p) = \mathbb{E}_p(E) - S(p)$, where $S(p)$ is the (Shannon) entropy of the distribution. Using variational methods, a remarkable result is easily shown (Wainwright and Jordan, 2008): minimizing \mathcal{F}_G over the set of all globally valid distributions (termed the *marginal polytope*) yields a value of $-\log Z$ at the true marginal distribution, given in (1).

This minimization is, however, computationally intractable, hence the approach of minimizing the Bethe free energy \mathcal{F} makes two approximations: (i) the marginal polytope is relaxed to the *local polytope*, where we require only *local* consistency, that is we deal with a *pseudo-marginal* vector q , which in our context may be considered $\{q_i = q(X_i = 1) \forall i \in \mathcal{V}, \mu_{ij} = q(x_i, x_j) \forall (i, j) \in \mathcal{E}\}$ subject to $q_i = \sum_{j \in \mathcal{N}(i)} \mu_{ij}, q_j = \sum_{i \in \mathcal{N}(j)} \mu_{ij} \forall i, j \in \mathcal{V}$; and (ii) the entropy S is approximated by the Bethe entropy $S_B = \sum_{(i,j) \in \mathcal{E}} S_{ij} + \sum_{i \in \mathcal{V}} (1 - d_i) S_i$, where S_{ij} is the entropy of μ_{ij} , S_i is the entropy of the singleton distribution and $d_i = |\mathcal{N}(i)|$ is the degree of i . The local polytope constraints imply that, given q_i and q_j ,

$$\mu_{ij} = \begin{pmatrix} 1 + \xi_{ij} - q_i - q_j & q_j - \xi_{ij} \\ q_i - \xi_{ij} & \xi_{ij} \end{pmatrix} \quad (2)$$

for some $\xi_{ij} \in [0, \min(q_i, q_j)]$, where $\mu_{ij}(a, b) = q(X_i = a, X_j = b)$. Hence, the global optimum of the Bethe free energy,

$$\begin{aligned} \mathcal{F}(q) &= \mathbb{E}_q(E) - S_B(q) \\ &= \sum_{(i,j) \in \mathcal{E}} -(W_{ij}\xi_{ij} + S_{ij}(q_i, q_j)) \\ &\quad + \sum_{i \in \mathcal{V}} (-\theta_i q_i + (d_i - 1)S_i(q_i)), \end{aligned} \quad (3)$$

is achieved by minimizing \mathcal{F} over the local polytope, with Z_B defined s.t. the result obtained equals $-\log Z_B$. See (Wainwright and Jordan, 2008) for details. Let $\alpha_{ij} = e^{W_{ij}} - 1$. $\alpha_{ij} = 0 \Leftrightarrow W_{ij} = 0$ may be assumed not to occur else the edge (i, j) may be deleted. α_{ij} has the same sign as W_{ij} , if positive then the edge (i, j) is *attractive*; if negative then the edge is *repulsive*. The MRF is attractive if all edges are attractive. As shown by Welling and Teh (2001), one can solve for ξ_{ij} explicitly in terms of q_i and q_j by minimizing \mathcal{F} , leading to a quadratic with real roots,

$$\alpha_{ij}\xi_{ij}^2 - [1 + \alpha_{ij}(q_i + q_j)]\xi_{ij} + (1 + \alpha_{ij})q_i q_j = 0. \quad (4)$$

For $\alpha_{ij} > 0$, $\xi_{ij}(q_i, q_j)$ is the lower root, for $\alpha_{ij} < 0$ it is the higher. Thus we may consider the minimization of \mathcal{F} over $q = (q_1, \dots, q_n) \in [0, 1]^n$. Collecting the pairwise terms of \mathcal{F} from (3) for one edge, define

$$f_{ij}(q_i, q_j) = -W_{ij}\xi_{ij}(q_i, q_j) - S_{ij}(q_i, q_j). \quad (5)$$

We are interested in *discretized pseudo-marginals* where for each q_i , we restrict its possible values to a discrete mesh \mathcal{M}_i of points in $[0, 1]$. The points may be spaced unevenly and we may have $\mathcal{M}_i \neq \mathcal{M}_j$. Let $N_i = |\mathcal{M}_i|$, and define $N = \sum_{i \in \mathcal{V}} N_i$ and $\Pi = \prod_{i \in \mathcal{V}} N_i$, the sum and product respectively of the number of mesh points in each dimension. Write \mathcal{M} for the entire mesh. Let \hat{q} be the location of a global optimum of \mathcal{F} . We say that a mesh construction $\mathcal{M}(\epsilon)$ is *sufficient* if, given $\epsilon > 0$, it can be guaranteed that \exists a mesh point $q^* \in \prod_{i \in \mathcal{V}} \mathcal{M}_i$ s.t. $\mathcal{F}(q^*) - \mathcal{F}(\hat{q}) \leq \epsilon$. The resulting discrete optimization problem may be framed as MAP inference in a multi-label MRF, where variable i takes values in \mathcal{M}_i , with the same topology (see §5).

2.1 INPUT MODEL SPECIFICATION

To be consistent with Welling and Teh (2001) and Weller and Jebara (2013a), for all theoretical analysis in this paper, we assume the reparameterization in (1). However, when an input model is specified, in order to avoid bias, we use singleton terms θ_i as in (1), but instead use pairwise energy terms given by $-\frac{W_{ij}}{2}x_i x_j - \frac{W_{ij}}{2}(1 - x_i)(1 - x_j)$. With this form, varying W_{ij} simply alters the degree of association between i and j . We assume maximum possible values W and T are known with $|\theta_i| \leq T \forall i \in \mathcal{V}$, and $|W_{ij}| \leq W \forall (i, j) \in \mathcal{E}$. The required transformation to convert from input model to the format of (1), simply takes $\theta_i \leftarrow \theta_i - \sum_{j \in \mathcal{N}(i)} W_{ij}/2$, leaving W_{ij} unaffected.

2.2 SUBMODULARITY

If all pairwise cost functions f_{ij} over $\mathcal{M}_i \times \mathcal{M}_j$ from (5) are submodular³, then the global discretized optimum may be found efficiently using graph cuts (Schlesinger and Flach, 2006). We require the following earlier result.

Theorem 1 (Submodularity for any discretization of an attractive model, see Weller and Jebara (2013a) Theorem 8, Korč et al. (2012)). *In a binary pairwise MRF, if an edge (i, j) is attractive, i.e. $W_{ij} > 0$, then the discretized multi-label MRF for any mesh \mathcal{M} is submodular for that edge. Hence if the MRF is fully attractive, then the discretized multi-label MRF is fully submodular for any discretization.*

2.3 FLIPPING VARIABLES

A useful technique for our analysis is to consider a model where some variables are flipped, i.e. given a model on $\{X_i\}$, consider a new model on $\{X'_i\}$ where $X'_i = 1 - X_i$ for some $i \in \mathcal{V}$. New model parameters $\{\theta'_i, W'_{ij}\}$ may be identified as in (Weller and Jebara, 2013a, §3) to preserve

³Here a pairwise multi-label function on a set of ordered labels $X_{ij} = \{1, \dots, K_i\} \times \{1, \dots, K_j\}$ is *submodular* iff $\forall x, y \in X_{ij}, f(x \wedge y) + f(x \vee y) \leq f(x) + f(y)$, where for $x = (x_1, x_2)$ and $y = (y_1, y_2)$, $(x \wedge y) = (\min(x_1, y_1), \min(x_2, y_2))$ and $(x \vee y) = (\max(x_1, y_1), \max(x_2, y_2))$. For binary variables this is equivalent to the edge potential being attractive.

energies of all states up to a constant. If all variables are flipped, new parameters are given by

$$W'_{ij} = W_{ij}, \theta'_i = -\theta_i - \sum_{j \in \mathcal{N}(i)} W_{ij}. \quad (6)$$

If the original model was attractive, so too is the new. If only a subset $\mathcal{R} \subseteq \mathcal{V}$ is flipped, let $X'_i = 1 - X_i$ if $i \in \mathcal{R}$, else $X'_i = X_i$ for $i \in \mathcal{S}$, where $\mathcal{S} = \mathcal{V} \setminus \mathcal{R}$. Let $\mathcal{E}_t = \{\text{edges with exactly } t \text{ ends in } \mathcal{R}\}$ for $t = 0, 1, 2$. Then we obtain

$$W'_{ij} = \begin{cases} W_{ij} & (i, j) \in \mathcal{E}_0 \cup \mathcal{E}_2, \\ -W_{ij} & (i, j) \in \mathcal{E}_1, \end{cases} \quad (7)$$

$$\theta'_i = \begin{cases} \theta_i + \sum_{(i,j) \in \mathcal{E}_1} W_{ij} & i \in \mathcal{S}, \\ -\theta_i - \sum_{(i,j) \in \mathcal{E}_2} W_{ij} & i \in \mathcal{R}. \end{cases}$$

Lemma 2. *Flipping variables changes affected pseudo-marginal matrix entries' locations but not values. \mathcal{F} is unchanged up to a constant, hence the locations of stationary points are unaffected. Proof in Weller and Jebara (2013a)*

2.4 PRELIMINARY BOUNDS

We use the following earlier results.

Lemma 3 (Weller and Jebara (2013a) Lemma 2). $\alpha_{ij} \geq 0 \Rightarrow \xi_{ij} \geq q_i q_j, \alpha_{ij} \leq 0 \Rightarrow \xi_{ij} \leq q_i q_j$.

Lemma 4 (Upper bound for ξ_{ij} for an attractive edge, Weller and Jebara (2013a) Lemma 6). *If $\alpha_{ij} > 0$, then $\xi_{ij} - q_i q_j \leq \frac{\alpha_{ij} m(1-M)}{1+\alpha_{ij}}$, where $m = \min(q_i, q_j)$ and $M = \max(q_i, q_j)$.*

Theorem 5 (Weller and Jebara (2013a) Theorem 4). *For general edge types (associative or repulsive), let $W_i = \sum_{j \in \mathcal{N}(i): W_{ij} > 0} W_{ij}$, $V_i = -\sum_{j \in \mathcal{N}(i): W_{ij} < 0} W_{ij}$. At any stationary point of the Bethe free energy, $\sigma(\theta_i - V_i) \leq q_i \leq \sigma(\theta_i + W_i)$, where $\sigma(x) = 1/(1 + \exp(-x))$ (sigmoid).*

For the efficiency of our overall approach, it is very desirable to tighten these bounds on locations of minima of \mathcal{F} since this both reduces the search space and allows a lower density of discretizing points in the mesh. For our theoretical results, we do not assume this can be done but in practice, it can be attempted efficiently by running either of the following two algorithms: Bethe bound propagation (BBP) from (Weller and Jebara, 2013a, §6), or using the approach from Mooij and Kappen (2007) which we term MK. Either method can achieve striking results quickly, though MK is our preferred method⁴ - this considers cavity fields around each variable and determines the range of possible beliefs after iterating LBP, starting from any initial values; since any minimum of \mathcal{F} corresponds to a fixed point of LBP (Yedidia et al., 2001), this bounds all minima.

⁴Both BBP and MK are anytime methods that converge quickly, and can be implemented such that each iteration runs in $O(m)$ time. MK takes a little longer but can yield tighter bounds.

Let the lower bounds obtained for q_i and $1 - q_i$ respectively be A_i and B_i so that $A_i \leq q_i \leq 1 - B_i$, and let the *Bethe box* be the orthotope given by $\prod_{i \in \mathcal{V}} [A_i, 1 - B_i]$. Define $\eta_i = \min(A_i, B_i)$, i.e. the closest that q_i can come to the extreme values of 0 or 1.

2.5 DERIVATIVES OF \mathcal{F}

Welling and Teh (2001) derived first partial derivatives of the Bethe free energy as

$$\frac{\partial \mathcal{F}}{\partial q_i} = -\theta_i + \log Q_i, \quad (8)$$

$$\text{where } Q_i = \frac{(1 - q_i)^{d_i - 1}}{q_i^{d_i - 1}} \frac{\prod_{j \in \mathcal{N}(i)} (q_i - \xi_{ij})}{\prod_{j \in \mathcal{N}(i)} (1 + \xi_{ij} - q_i - q_j)}.$$

Weller and Jebara (2013a) derived all second partial derivatives.

Theorem 6 (All terms of the Hessian, see Weller and Jebara (2013a) §4.3 and Lemma 9). *Let H be the Hessian of \mathcal{F} for a binary pairwise model, i.e. $H_{ij} = \frac{\partial^2 \mathcal{F}}{\partial q_i \partial q_j}$, and d_i be the degree of variable X_i , then*

$$H_{ii} = -\frac{d_i - 1}{q_i(1 - q_i)} + \sum_{j \in \mathcal{N}(i)} \frac{q_j(1 - q_j)}{T_{ij}} \geq \frac{1}{q_i(1 - q_i)},$$

$$H_{ij} = \begin{cases} \frac{q_i q_j - \xi_{ij}}{T_{ij}} & (i, j) \in \mathcal{E} \\ 0 & (i, j) \notin \mathcal{E}, i \neq j, \end{cases}$$

$$\text{where } T_{ij} = q_i q_j (1 - q_i)(1 - q_j) - (\xi_{ij} - q_i q_j)^2 \geq 0 \text{ with equality iff } q_i \text{ or } q_j \in \{0, 1\}. \quad (9)$$

3 NEW APPROACH: GRADMESH

We develop a new approach to constructing a sufficient mesh \mathcal{M} by analyzing bounds on the first derivatives of \mathcal{F} . To help distinguish between methods, we call the new first derivative approach *gradMesh*, and the earlier, second derivative approach *curvMesh*. The new gradMesh approach yields several attractive features:

- For attractive models, we obtain a FPTAS with worst case runtime $O(\epsilon^{-3} n^3 m^3 W^3)$ and no restriction on topology, unlike earlier work (Weller and Jebara, 2013a) which required max degree $\Delta = O(\log n)$.
- Our sufficient mesh is typically dramatically coarser than the earlier method of Weller and Jebara (2013a) unless ϵ is very small, leading to a much smaller subsequent MAP problem. Here, the sum of the number of discretizing points in each dimension, $N = O\left(\frac{nmW}{\epsilon}\right)$. For comparison, the earlier method, even after our improvements in §4, forms a mesh with $N = O\left(\epsilon^{-1/2} n^{7/4} \Delta^{3/4} \exp\left[\frac{1}{2}(W(1 + \Delta/2) + T)\right]\right)$. See §6.1 for examples.

- The approach immediately handles a general model with both attractive and repulsive edges. Hence approximating $\log Z_B$ may be reduced to a discrete multi-label MAP inference problem. This is valuable due to the availability of many MAP techniques, see §5.

First consider a model which is fully attractive around variable X_i , i.e. $W_{ij} > 0 \forall j \in \mathcal{N}(i)$. From (8) and Lemma 3, we obtain

$$\frac{\partial \mathcal{F}}{\partial q_i} = -\theta_i + \log Q_i \leq -\theta_i + \log \frac{q_i}{1 - q_i}. \quad (10)$$

Flip all variables (see §2.3). Write $'$ for the parameters of the new flipped model, which is also fully attractive, then using (6) and (10),

$$\begin{aligned} \frac{\partial \mathcal{F}'}{\partial q'_i} &\leq -\theta'_i + \log \frac{q'_i}{1 - q'_i} \\ \Leftrightarrow -\theta_i - W_i + \log \frac{q_i}{1 - q_i} &\leq \frac{\partial \mathcal{F}}{\partial q_i}. \end{aligned}$$

Combining this with (10) yields the sandwich result

$$-\theta_i - W_i + \log \frac{q_i}{1 - q_i} \leq \frac{\partial \mathcal{F}}{\partial q_i} \leq -\theta_i + \log \frac{q_i}{1 - q_i}.$$

Now generalize to consider the case that i has some neighbors \mathcal{R} to which it is adjacent by repulsive edges. In this case, flip those nodes \mathcal{R} (see §2.3) to yield a model, which we denote by $''$, which is fully attractive around i , hence we may apply the above result. By (7) we have $\theta''_i = \theta_i - V_i$, and using $W''_i = W_i + V_i$, we obtain that for a general model,

$$-\theta_i - W_i + \log \frac{q_i}{1 - q_i} \leq \frac{\partial \mathcal{F}}{\partial q_i} \leq -\theta_i + V_i + \log \frac{q_i}{1 - q_i}. \quad (11)$$

This bounds each first derivative $\frac{\partial \mathcal{F}}{\partial q_i}$ within a range of width $V_i + W_i = \sum_{j \in \mathcal{N}(i)} |W_{ij}|$, which is sufficient for the main theoretical result, see (15). We take the opportunity, however, to describe a method which sometimes significantly narrows this range, thereby improving the result in practice.

Using one $O(m)$ iteration of the belief propagation algorithm (BBP) derived in (Weller and Jebara, 2013a, Supplement), allows us to refine the bounds for variable X_i of (11) based on the $[A_j, 1 - B_j]$ location bounds on its neighbors $j \in \mathcal{N}(i)$, to show

$$\begin{aligned} f_i^L(q_i) &\leq \frac{\partial \mathcal{F}}{\partial q_i} \leq f_i^U(q_i), \text{ where} \\ f_i^L(q_i) &= -\theta_i - W_i + \log \frac{q_i}{1 - q_i} + \log U_i \\ f_i^U(q_i) &= -\theta_i + V_i + \log \frac{q_i}{1 - q_i} - \log L_i. \end{aligned} \quad (12)$$

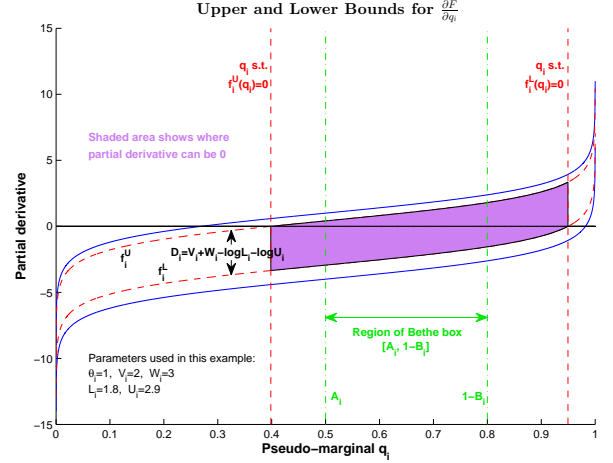


Figure 1: Upper and Lower Bounds for $\frac{\partial \mathcal{F}}{\partial q_i}$. Solid blue curves show worst case bounds (11) as functions of q_i , and are different by a constant $V_i + W_i = \sum_{j \in \mathcal{N}(i)} |W_{ij}|$. Dashed red curves show the upper $f_i^U(q_i)$ and lower $f_i^L(q_i)$ bounds (12) after being lowered by $\log L_i$ and raised by $\log U_i$ respectively, which incorporate the information from the bounds of neighboring variables. All bounding curves are strictly monotonic. The Bethe box region for q_i must lie within the shaded region demarcated by vertical red dashed lines, but we may have better bounds available, e.g. from MK, as shown by A_i and $1 - B_i$.

L_i, U_i are each > 1 with $\log L_i + \log U_i \leq V_i + W_i$. They are computed as $L_i = \prod_{j \in \mathcal{N}(i)} L_{ij}$, $U_i = \prod_{j \in \mathcal{N}(i)} U_{ij}$,

$$\begin{aligned} \text{with } L_{ij} &= \begin{cases} 1 + \frac{\alpha_{ij} A_j}{1 + \alpha_{ij} (1 - B_i)(1 - A_j)} & \text{if } W_{ij} > 0 \\ 1 + \frac{\alpha_{ij} B_j}{1 + \alpha_{ij} (1 - B_i)(1 - B_j)} & \text{if } W_{ij} < 0 \end{cases} \\ U_{ij} &= \begin{cases} 1 + \frac{\alpha_{ij} B_j}{1 + \alpha_{ij} (1 - A_i)(1 - B_j)} & \text{if } W_{ij} > 0 \\ 1 + \frac{\alpha_{ij} A_j}{1 + \alpha_{ij} (1 - A_i)(1 - A_j)} & \text{if } W_{ij} < 0 \end{cases} \end{aligned}$$

See Figure 1 for an example. We make the following observations:

- The upper bound is equal to the lower bound plus the constant $D_i = V_i + W_i - \log L_i - \log U_i \geq 0$.
- The bound curves are monotonically increasing with q_i , ranging from $-\infty$ to $+\infty$ as q_i ranges from 0 to 1.
- A necessary condition to be within the Bethe box is that the upper bound is ≥ 0 and the lower bound is ≤ 0 . Hence, anywhere within the Bethe box, we must have bounded derivative, $|\frac{\partial \mathcal{F}}{\partial q_i}| \leq D_i$. BBP generates $\{[A_i, 1 - B_i]\}$ bounds by iteratively updating with L_i, U_i terms. In general, however, we may have better bounds from any other method, such as MK, which lead to higher L_i and U_i parameters and lower D_i .

\mathcal{F} is continuous on $[0, 1]^n$ and differentiable everywhere in $(0, 1)^n$ with partial derivatives satisfying (12). $f_i^L(q_i)$ and $f_i^U(q_i)$ are continuous and integrable. Indeed, using the

notation $[\phi(x)]_{x=a}^{x=b} = \phi(b) - \phi(a)$,

$$\int_a^b \log \frac{q_i}{1-q_i} dq_i = \left[q_i \log q_i + (1-q_i) \log(1-q_i) \right]_{q_i=a}^{q_i=b} \quad (13)$$

for $0 \leq a \leq b \leq 1$, which relates to the binary entropy function $H(p) = -p \log p - (1-p) \log(1-p)$, recall the definition of \mathcal{F} . We remark that although $\frac{\partial \mathcal{F}}{\partial q_i}$ tends to $-\infty$ or $+\infty$ as q_i tends to 0 or 1, the integral converges (taking $0 \log 0 = 0$).

Hence if $\hat{q} = (\hat{q}_1, \dots, \hat{q}_n)$ is the location of a global minimum, then for any $q = (q_1, \dots, q_n)$ in the Bethe box,

$$\mathcal{F}(q) - \mathcal{F}(\hat{q}) \leq \sum_{i: \hat{q}_i \leq q_i} \int_{\hat{q}_i}^{q_i} f_i^U(q_i) dq_i + \sum_{i: q_i < \hat{q}_i} \int_{q_i}^{\hat{q}_i} -f_i^L(q_i) dq_i. \quad (14)$$

To construct a sufficient mesh, a simple initial bound relies on $|\frac{\partial \mathcal{F}}{\partial q_i}| \leq D_i$. If mesh points \mathcal{M}_i are chosen s.t. in dimension i there must be a point q^* within γ_i of a global minimum (which can be achieved using a mesh width in each dimension of $2\gamma_i$), then by setting $\gamma_i = \frac{\epsilon}{nD_i}$, we obtain $\mathcal{F}(q^*) - \mathcal{F}(\hat{q}) \leq \sum_i D_i \frac{\epsilon}{nD_i} = \epsilon$. It is easily seen that $N_i \leq 1 + \lceil \frac{1}{2\gamma_i} \rceil$, hence the total number of mesh points, $N = \sum_{i \in \mathcal{V}} N_i$, satisfies

$$\begin{aligned} N &\leq 2n + \frac{n}{2\epsilon} \sum_i D_i \leq 2n + \frac{n}{\epsilon} \sum_{(i,j) \in \mathcal{E}} |W_{ij}| \\ &= O\left(\frac{n}{\epsilon} \sum_{(i,j) \in \mathcal{E}} |W_{ij}|\right) = O\left(\frac{nmW}{\epsilon}\right), \end{aligned} \quad (15)$$

since $D_i \leq V_i + W_i = \sum_{j \in \mathcal{N}(i)} |W_{ij}|$. Here $W = \max_{(i,j) \in \mathcal{E}} |W_{ij}|$ and $m = |\mathcal{E}|$ is the number of edges.

If the initial model is fully attractive, then by Theorem 1 we obtain a submodular multi-label MAP problem which is solvable using graph cuts with worst case runtime $O(N^3) = O(\epsilon^{-3} n^3 m^3 W^3)$ (Schlesinger and Flach, 2006; Greig et al., 1989; Goldberg and Tarjan, 1988).

Note from the first expression in (15) that if we have information on individual edge weights then we have a better bound using $\sum_{(i,j) \in \mathcal{E}} |W_{ij}|$ rather than just mW .

For comparison, the earlier second derivative approach of Weller and Jebara (2013a) has runtime $O(\epsilon^{-\frac{3}{2}} n^6 \Sigma^{\frac{3}{2}} \Omega^{\frac{3}{2}})$, where, even using the improved method in §4 here, $\Omega = O(\Delta e^{W(1+\Delta/2)+T})$. Unless ϵ is very small, the new first derivative approach is typically dramatically more efficient and more useful in practice. Further, it naturally handles both attractive and repulsive edge weights in the same way.

3.1 REFINEMENTS, ADAPTIVE METHODS

Since the resulting multi-label MAP inference problem (which is not submodular in general) is NP-hard (Shimony,

1994), it is helpful to minimize its size. As noted above, setting $\gamma_i = \frac{\epsilon}{nD_i}$, which we term the *simple method*, yields a sufficient mesh, where $|\frac{\partial \mathcal{F}}{\partial q_i}| \leq D_i = V_i + W_i - \log L_i - \log U_i$. However, since the bounding curves are monotonic with $f_i^U \geq 0$ and $f_i^L \leq 0$, a better bound for the magnitude of the derivative is available by setting $D_i = \max\{f_i^U(1 - B_i), -f_i^L(A_i)\}$.

3.1.1 The minsum Method

We define N_i = the number of mesh points in dimension i , with sum $N = \sum_{i \in \mathcal{V}} N_i$ and product $\Pi = \prod_{i \in \mathcal{V}} N_i$. For a fully attractive model, the resulting MAP problem may be solved in time $O(N^3)$ by graph cuts (Theorem 1, (Schlesinger and Flach, 2006; Greig et al., 1989; Goldberg and Tarjan, 1988)), so it is sensible to minimize N . In other cases, however, it is less clear what to minimize. For example, a brute force search over all points would take time $\Theta(\Pi)$.

Define the spread of possible values in dimension i as $S_i = 1 - B_i - A_i$ and note $N_i = 1 + \lceil \frac{S_i}{2\gamma_i} \rceil$ is required to cover the whole range. To minimize N while ensuring the mesh is sufficient, consider the Lagrangian $\mathcal{L} = \sum_{i \in \mathcal{V}} \frac{S_i}{2\gamma_i} - \lambda(\epsilon - \sum_{i \in \mathcal{V}} \gamma_i D_i)$, where D_i is set as in the simple method (§3.1). Optimizing gives

$$\gamma_i = \frac{\epsilon}{\sum_{j \in \mathcal{V}} \sqrt{S_j D_j}} \sqrt{\frac{S_i}{D_i}}, \text{ and } N \leq 2n + \frac{1}{2\epsilon} \left(\sum_{i \in \mathcal{V}} \sqrt{S_i D_i} \right)^2 \quad (16)$$

which we term the *minsum method*. Note $D_i \leq d_i W$ where d_i is the degree of X_i , hence $(\sum_{i \in \mathcal{V}} \sqrt{S_i D_i})^2 \leq W (\sum_{i \in \mathcal{V}} \sqrt{d_i})^2$. By Cauchy-Schwartz and the handshake lemma, $(\sum_{i \in \mathcal{V}} \sqrt{d_i})^2 \leq n \sum_{i \in \mathcal{V}} d_i = 2mn$, with equality iff the d_i are constant, i.e. the graph is regular.

If instead Π is minimized, rather than N , a similar argument shows that the simple method (§3.1) is optimal.

3.1.2 Adaptive Methods

The previous methods rely on one bound D_i for $|\frac{\partial \mathcal{F}}{\partial q_i}|$ over the whole range $[A_i, 1 - B_i]$. However, we may increase efficiency by using local bounds to vary the mesh width across the range. A bound on the maximum magnitude of the derivative over any sub-range may be found by checking just $-f_i^L$ at the lower end and f_i^U at the upper end.

This may be improved by using the exact integral as in (14). First, constant proportions $k_i > 0$ should be chosen with $\sum_i k_i = 1$. Next, the first or smallest mesh point $\gamma_1^i \in \mathcal{M}_i$ should be set s.t. $\int_{A_i}^{\gamma_1^i} f_i^U(q_i) dq_i = k_i \epsilon$. This will ensure that γ_1^i covers all points to its left in the sense that $\mathcal{F}[q_i = \gamma_1^i] - \mathcal{F}[q_i \in [A_i, \gamma_1^i]] \leq k_i \epsilon$ where all other variables $q_j, j \neq i$, are held constant at any values within the Bethe box. γ_1^i also covers all points to its right up to what we term

its *reach*, i.e. the point r_1^i s.t. $\int_{\gamma_1^i}^{r_1^i} -f_i^L(q_i)dq_i = k_i\epsilon$. Next, γ_2^i is chosen as before, using r_1^i as the left extreme rather than A_i , and so on, until the final mesh point is computed with $\text{reach} \geq 1 - B_i$. This yields an optimal mesh for the choice of $\{k_i\}$.

If $k_i = \frac{1}{n}$, we achieve an optimized *adaptive simple* method. If $k_i = \frac{\sqrt{S_i D_i}}{\sum_{j \in \mathcal{V}} \sqrt{S_j D_j}}$, we achieve an *adaptive minsum* method. For many problems, this adaptive minsum method will be the most efficient.

Integrals are easily computed using (13). To our knowledge, computing optimal points $\{\gamma_s^i\}$ is not possible analytically, but each may be found with high accuracy in just a few iterations using a search method, hence total time to compute the mesh is $O(N)$, which is negligible compared to solving the subsequent MAP problem.

4 REVISITING THE SECOND DERIVATIVE APPROACH: CURVMESH

We shall review and then refine the second derivative approach used in (Weller and Jebara, 2013a, §5), which we call *curvMesh*. Its mesh size (measured by N , the total number of points summed over the dimensions) grows as $O(\epsilon^{-1/2})$ rather than as $O(\epsilon^{-1})$ in the new first derivative gradMesh approach. In practice, however, unless ϵ is very small, gradMesh is much more efficient (see Figure 2).

As in this paper, the possible location of a global minimum \hat{q} was first bounded in the Bethe box given by $\prod_{i \in \mathcal{V}} [A_i, 1 - B_i]$. Next an upper bound Λ was derived on the maximum possible eigenvalue of the Hessian H of \mathcal{F} anywhere within the Bethe box, where it was required that all edges be attractive. Then a mesh of constant width in every dimension was introduced s.t. the nearest mesh point q^* to \hat{q} was at most γ away in each dimension. Hence the ℓ_2 distance δ satisfies $\delta^2 \leq n\gamma^2$ and by Taylor's theorem, $F(q^*) \leq F(\hat{q}) + \frac{1}{2}\Lambda\delta^2$. Λ was computed by bounding the maximum magnitude of any element of H . Considering Theorem 6, this involves separate analysis of diagonal H_{ii} terms, which are positive and were bounded above by the term b ; and edge H_{ij} terms, which are negative for attractive edges, whose magnitude was bounded above by a . Then Ω was set as $\max(a, b)$, and Σ as the proportion of non-zero entries in H . Finally, $\Lambda \leq \sqrt{\text{tr}(H^T H)} \leq \sqrt{\Sigma n^2 \Omega^2} = n\Omega\sqrt{\Sigma}$.

4.1 IMPROVED BOUND FOR AN ATTRACTIVE MODEL

We improve the upper bound for Λ by improving the a bound for attractive edges to derive \tilde{a} , a better upper bound on $-H_{ij}$. Essentially, a more careful analysis allows a po-

tentially small term in the numerator and denominator to be canceled before bounding. Writing $\bar{\eta} = \min_{i \in \mathcal{V}} \eta_i(1 - \eta_i)$, i.e. the closest that any dimension can come to 0 or 1, the result is that

$$\begin{aligned} -H_{ij} &\leq \left(\frac{\alpha_{ij}}{1 + \alpha_{ij}} \right) / \bar{\eta} \left(1 - \left(\frac{\alpha_{ij}}{1 + \alpha_{ij}} \right)^2 \right) \\ &= O(e^{W(1+\Delta/2)+T}). \end{aligned} \quad (17)$$

Thus, $\tilde{a} = O(e^{W(1+\Delta/2)+T})$ which compares favorably to the earlier bound in Weller and Jebara (2013a), where $a = O(e^{W(1+\Delta)+2T})$. Recall $b = O(\Delta e^{W(1+\Delta/2)+T})$ and $\Omega = \max(a, b)$, so using the new \tilde{a} bound, now $\Omega = O(\Delta e^{W(1+\Delta/2)+T})$. Details and derivations are in the supplement.

4.2 EXTENDING TO A GENERAL MODEL

Using flipping arguments from §2.3, we are able to extend the method of Weller and Jebara (2013a) to apply to general (non-attractive) models. Interestingly, the bounds derived for $\Omega = \max(a, b)$ take exactly the same form as for the purely attractive case, except that now $-W \leq W_{ij} \leq W$, whereas previously it was required that $0 \leq W_{ij} \leq W$. Details and derivations are in the supplement.

5 RESULTING MULTI-LABEL MAP

After computing a sufficient mesh, it remains to solve the multi-label MAP inference problem on a MRF with the same topology as the initial model, where each q_i takes values in \mathcal{M}_i . In general, this is NP-hard (Shimony, 1994).

5.1 TRACTABLE CASES

If it happens that all cost functions are submodular (as is always the case if the initial model is fully attractive by Theorem 1), then as already noted, it may be solved efficiently using graph cut methods, which rely on solving a max flow/min cut problem on a related graph, with worst case runtime $O(N^3)$ (Schlesinger and Flach, 2006; Greig et al., 1989; Goldberg and Tarjan, 1988). Using the algorithm of Boykov and Kolmogorov (2004), performance is typically much faster, sometimes approaching $O(N)$. This submodular setting is the only known class of problem which is solvable for any topology.

Alternatively, the topological restriction of bounded tree-width allows tractable inference (Pearl, 1988). Further, under mild assumptions, this was shown to be the only restriction which will allow efficient inference for any cost functions (Chandrasekaran et al., 2008). We note that if the problem has bounded tree-width, then so too does the original binary pairwise model, hence exact inference (to yield the true marginals or the true partition function Z) on

the original model is tractable using the junction tree algorithm, making our approximation result less interesting for this class. In contrast, although MAP inference is tractable for any attractive binary pairwise model, marginal inference and computing Z are not (Jerrum and Sinclair, 1993).

A recent approach reducing MAP inference to identifying a maximum weight stable set in a derived weighted graph (Jebara, 2014; Weller and Jebara, 2013b) shows promise, allowing efficient inference if the derived graph is perfect. Further, testing if this graph is perfect can be performed in polynomial time (Jebara, 2014; Chudnovsky et al., 2005).

5.2 INTRACTABLE MAP CASES

Many different methods are available, see Kappes et al. (2013) for a recent survey. Some, such as dual approaches, may provide a helpful bound even if the optimum is not found. Indeed, a LP relaxation will run in polynomial time and return an upper bound on $\log Z_B$ that may be useful. A lower bound may be found from any discrete point, and this may be improved using local search methods.

Note that the Bethe box bounds on each $q_i \in [A_i, 1 - B_i]$ are worst case, irrespective of other variables. However, given a particular value for one or more q_j , $j \in \mathcal{N}(i)$, either BBP (Weller and Jebara, 2013a, §6) or MK (Mooij and Kappen, 2007) can produce better bounds on q_i , which may be helpful for pruning the solution space.

5.2.1 Persistent partial optimization approaches

The multi-label implementation of quadratic pseudo-Boolean optimization (Kohli et al., 2008, MQPBO), and the method of Kovtun (2003), are examples of this class. Both consider LP-relaxations and run in polynomial time. In our context, the output consists of ranges (which in the best case could be one point) of settings for some subset of the variables. If any such ranges are returned, the strong persistence property ensures that *any* MAP solution satisfies the ranges. Hence, these may be used to update $\{A_i, B_i\}$ bounds (padding the discretized range to the full continuous range covered by the end points if needed), compute a new, smaller, sufficient mesh and repeat until no improvement is obtained.

6 EXPERIMENTS

6.1 COMPARISON TO EARLIER WORK

We compared the new mesh construction methods from this paper with the earlier approach by Weller and Jebara (2013a), see Figure 2. We considered two values of ϵ : 1 (medium resolution) and 0.1 (fine resolution). For each value, we generated random MRFs on n variables, all pairwise connected, where $\theta_i \sim U[-2, 2]$ and $W_{ij} \sim$

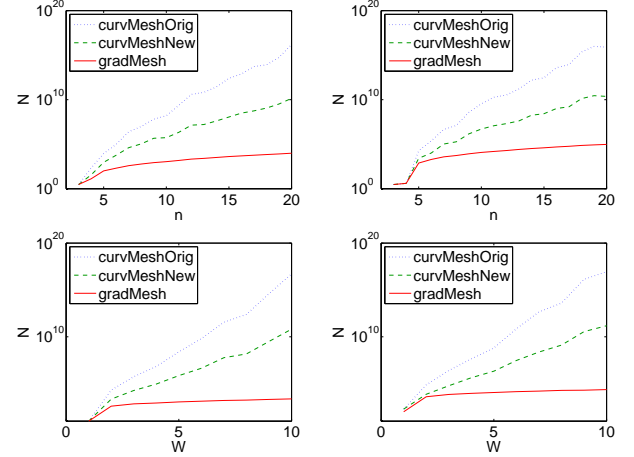


Figure 2: Variation in N = sum of number of mesh points in each dimension, **log scale**, as: (top) n = number of variables is changed, keeping $W = 5$ fixed; (bottom) W = maximum coupling strength is changed, keeping $n = 10$ fixed. On the left, $\epsilon = 1$ (medium resolution); on the right, $\epsilon = 0.1$ (fine resolution). In each case the topology is a complete graph, edge weights are chosen $W_{ij} \sim U[-W, W]$ and $\theta_i \sim U[-2, 2]$. Average over 10 random models for each value. *curvMeshOrig* is the original method of Weller and Jebara (2013a); *curvMeshNew* is our refinement, see §4; *gradMesh* is our new first derivative minsum method, see §3. For more details, see text of §6.1.

$U[-W, W]$, using the input convention of §2.1.⁵ We show results first for fixed $W = 5$ as n is varied from 3 to 20, then for fixed $n = 10$ as W is varied from 1 to 10, generating 10 random models for each value. Of the various first derivative gradMesh methods, only minsum is shown since the others would not be sufficiently distinguishable on these plots.⁶

Note that N is shown on a log axis, thus we observe that the new methods dramatically outperform that of Weller and Jebara (2013a) by many orders of magnitude for most cases of interest, even for small ϵ . Further, recall that $N = \sum_i N_i$ is the sum of the number of mesh points in each dimension. The runtime of the overall algorithm is certainly $\Omega(N)$, even for attractive models⁷, and for general models is typically a significantly higher power, thus further demonstrating the benefit of the new methods.

⁵The original method of Weller and Jebara (2013a) could only handle attractive models but we augment it as in §4.2. Plots for attractive models, where $W_{ij} \sim U[0, W]$ are very similar to those shown.

⁶In practice, the adaptive methods typically produce a mesh with about half the number of points in each dimension.

⁷In our experiments on attractive models, the Boykov-Kolmogorov algorithm typically runs in time $O(N^{1.5})$ to $O(N^{2.5})$.

6.2 POWER NETWORK

As a first step toward applying our algorithm to explore the usefulness of the global optimum of the Bethe approximation, here we consider one setting where LBP fails to converge, yet still we achieve reasonable results.

We aim to predict transformer failures in a power network (Rudin et al., 2012). Since the real data is sensitive, our experiments use synthetic data. Let $X_i \in \{0, 1\}$ indicate if transformer i has failed or not. Each transformer has a probability of failure on its own which is represented by a singleton potential θ_i . However, when connected in a network, a transformer can propagate its failure to nearby nodes (as in viral contagion) since the edges in the network form associative dependencies. We assume that homogeneous attractive pairwise potentials couple all transformers that are connected by an edge, i.e. $W_{ij} = W \forall (i, j) \in \mathcal{E}$. The network topology creates a Markov random field specifying the distribution $p(X_1, \dots, X_n)$. Our goal is to compute the marginal probability of failure of each transformer within the network (not simply in isolation as in Rudin et al. (2012)). Since recovering $p(X_i)$ is hard, we estimate Bethe pseudo-marginals $q_i = q(X_i = 1)$ through our algorithm, which emerge as the arg min when optimizing the Bethe free energy.

A single simulated sub-network of 55 connected transformers was generated using a random preferential attachment model, resulting in average degree 2 (see Figure 3 in the Appendix). Typical settings of $\theta_i = -2$ and $W = 4$ were specified (using the input model specification of §2.1). We attempted to run BP using the libDAI package (Mooij, 2010) but were unable to achieve convergence, even with multiple initial values, using various sequential or parallel settings and with damping. However, running our gradMesh adaptive minsum algorithm with $\epsilon = 1$ achieved reasonable results as shown in Table 1, where true values were obtained with the junction tree algorithm.

$\epsilon = 1$ PTAS for $\log Z_B$	Error from true value
Mean ℓ_1 error of single marginals	0.003
Log-partition function	0.26

Table 1: Results on simulated power network

It has been suggested that the Bethe approximation is poor when BP fails to converge (Mooij and Kappen, 2005). Our new method will allow this to be explored rigorously in future work. The initial result above is a promising first step and justifies further investigation.

7 DISCUSSION & FUTURE WORK

To our knowledge, we have derived the first ϵ -approximation algorithm for $\log Z_B$ for a general binary pairwise model. Our approaches are useful in practice, and

much more efficient than the earlier method of Weller and Jebara (2013a). From experiments run, we note that the ϵ bounds for the adaptive minsum first derivative *gradMesh* approach appear to be close to tight since we have found models where the optimum returned when run with $\epsilon = 1$ is more than 0.5 different to that for $\epsilon = 0.1$. When applied to attractive models, we guarantee a FPTAS with no degree restriction.

As described in §6.2, Bethe pseudo-marginals may be recovered from our approach by taking the q^* that is returned as the arg min of \mathcal{F} over the discrete mesh. However, although $\mathcal{F}(q^*)$ is guaranteed within ϵ of the optimum, there is no guarantee that q^* will necessarily be close to a true Bethe optimum pseudo-marginal. For example, the surface could be very flat over a wide region, or the true optimum might be $\frac{\epsilon}{2}$ better at a location far from q^* . We sketch out how our approach may be used to bound the location of a global optimum pseudo-marginal, though note that there is no runtime guarantee. First pick an initial ϵ_1 and run the main algorithm to find q_1^* . Now use any method to solve for the second best discretized mesh point q_2^* . If it happens that $\mathcal{F}(q_2^*) \geq \mathcal{F}(q_1^*) + \epsilon_1$ then, by the nature of the mesh construction, there must be a global minimum within the orthotope given by the neighboring mesh points of q_1^* in each dimension⁸ and we terminate. On the other hand, if $\mathcal{F}(q_2^*) < \mathcal{F}(q_1^*)$ then we reduce ϵ , for example to $\frac{\epsilon_1}{2}$ and repeat until we’re successful.

Future work includes further reducing the size of the mesh, considering how it should be selected to simplify the subsequent discrete optimization problem, and exploring applications. Importantly, we now have the opportunity to examine rigorously the performance of the global Bethe optimum. In addition, this will provide a benchmark against which to compare other (non-global) Bethe approaches that typically run more quickly, such as LBP or CCCP (Yuille, 2002). Another interesting avenue is to use our algorithm as a subroutine in a dual decomposition approach to optimize over a tighter relaxation of the marginal polytope.

Acknowledgements

We are grateful to Kui Tang for help with coding, and to David Sontag, Kui Tang, Nicholas Ruoizzi and Tomaž Slivnik for helpful discussions. This work was supported in part by NSF grants IIS-1117631 and CCF-1302269.

References

- D. Belanger, D. Sheldon, and A. McCallum. Marginal inference in MRFs using Frank-Wolfe. In *NIPS Workshop on Greedy Optimization, Frank-Wolfe and Friends*, December 2013.
- H. Bethe. Statistical theory of superlattices. *Proc. R. Soc. Lond. A*, 150(871):552–575, 1935.

⁸In fact, the optimum must be within a tighter orthotope based on the *reach* down and up, in each dimension, of q_1^* , see 3.1.2.

- Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(9):1124–1137, 2004.
- V. Chandrasekaran, N. Srebro, and P. Harsha. Complexity of inference in graphical models. In D. McAllester and P. Myllymäki, editors, *UAI*, pages 70–78. AUAI Press, 2008. ISBN 0-9749039-4-9.
- M. Chudnovsky, G. Cornuéjols, X. Liu, P. Seymour, and K. Vuskovic. Recognizing Berge graphs. *Combinatorica*, 25(2):143–186, 2005.
- A. Goldberg and R. Tarjan. A new approach to the maximum flow problem. *Journal of the ACM*, 35:921–940, 1988.
- D. Greig, B. Porteous, and A. Seheult. Exact maximum a posteriori estimation for binary images. *J. Royal Statistical Soc., Series B*, 51(2):271–279, 1989.
- T. Hazan and T. Jaakkola. On the partition function and random maximum a-posteriori perturbations. In *ICML*, 2012.
- U. Heinemann and A. Globerson. What cannot be learned with Bethe approximations. In *UAI*, pages 319–326, 2011.
- T. Jebara. *Tractability: Practical Approaches to Hard Problems*, chapter Perfect graphs and graphical modeling. Cambridge Press, 2014.
- M. Jerrum and A. Sinclair. Polynomial-time approximation algorithms for the Ising model. *SIAM J. Comput.*, 22(5):1087–1116, 1993.
- J. Kappes, B. Andres, F. Hamprecht, C. Schnörr, S. Nowozin, D. Batra, S. Kim, B. Kausler, J. Lellmann, N. Komodakis, and C. Rother. A comparative study of modern inference techniques for discrete energy minimization problems. In *CVPR*, 2013.
- P. Kohli, A. Shekhovtsov, C. Rother, V. Kolmogorov, and P. Torr. On partial optimality in multi-label MRFs. In W. Cohen, A. McCallum, and S. Roweis, editors, *ICML*, volume 307 of *ACM International Conference Proceeding Series*, pages 480–487. ACM, 2008. ISBN 978-1-60558-205-4.
- F. Korč, V. Kolmogorov, and C. Lampert. Approximating marginals using discrete energy minimization. Technical report, IST Austria, 2012.
- I. Kovtun. Partial optimal labeling search for a NP-hard subclass of (max, +) problems. In B. Michaelis and G. Krell, editors, *DAGM-Symposium*, volume 2781 of *Lecture Notes in Computer Science*, pages 402–409. Springer, 2003. ISBN 3-540-40861-4.
- R. McEliece, D. MacKay, and J. Cheng. Turbo decoding as an instance of Pearl’s “Belief Propagation” algorithm. *IEEE Journal on Selected Areas in Communications*, 16(2):140–152, 1998.
- O. Meshi, A. Jaimovich, A. Globerson, and N. Friedman. Convexifying the Bethe free energy. In *UAI*, pages 402–410, 2009.
- J. Mooij. libDAI: A free and open source C++ library for discrete approximate inference in graphical models. *Journal of Machine Learning Research*, 11:2169–2173, August 2010. URL <http://www.jmlr.org/papers/volume11/mooij10a/mooij10a.pdf>.
- J. Mooij and H. Kappen. Sufficient conditions for convergence of loopy belief propagation. In *Proceedings of the 21st Annual Conference on Uncertainty in Artificial Intelligence (UAI-05)*, pages 396–403. AUAI Press, 2005.
- J. Mooij and H. Kappen. Sufficient conditions for convergence of the sum-product algorithm. *IEEE Transactions on Information Theory*, 53(12):4422–4437, December 2007.
- K. Murphy, Y. Weiss, and M. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Uncertainty in Artificial Intelligence (UAI)*, 1999.
- J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.
- C. Rudin, D. Waltz, R. Anderson, A. Boulanger, A. Salieb-Aouissi, M. Chow, H. Dutta, P. Gross, B. Huang, and S. Ierome. Machine learning for the New York City power grid. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(2):328–345, February 2012. ISSN 0162-8828. doi: 10.1109/TPAMI.2011.108.
- D. Schlesinger and B. Flach. Transforming an arbitrary minsum problem into a binary one. Technical report, Dresden University of Technology, 2006.
- S. Shimony. Finding MAPs for belief networks is NP-hard. *Artificial Intelligence*, 68(2):399–410, 1994.
- J. Shin. Complexity of Bethe approximation. In *Artificial Intelligence and Statistics*, 2012.
- M. Wainwright and M. Jordan. Graphical models, exponential families and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.
- A. Weller and T. Jebara. Bethe bounds and approximating the global optimum. In *Artificial Intelligence and Statistics*, 2013a.
- A. Weller and T. Jebara. On MAP inference by MWSS on perfect graphs. In *Uncertainty in Artificial Intelligence (UAI)*, 2013b.
- M. Welling and Y. Teh. Belief optimization for binary networks: A stable alternative to loopy belief propagation. In *Uncertainty in Artificial Intelligence (UAI)*, 2001.
- J. Yedidia, W. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. In *International Joint Conference on Artificial Intelligence, Distinguished Lecture Track*, 2001.
- A. Yuille. CCCP algorithms to minimize the Bethe and Kikuchi free energies: Convergent alternatives to belief propagation. *Neural Computation*, 14:1691–1722, 2002.

Understanding the Bethe Approximation: When and How can it go Wrong?

Adrian Weller

Columbia University
New York NY 10027

adrian@cs.columbia.edu

Kui Tang

Columbia University
New York NY 10027

kt2384@cs.columbia.edu

David Sontag

New York University
New York NY 10012

dsontag@cs.nyu.edu

Tony Jebara

Columbia University
New York NY 10027

jebara@cs.columbia.edu

Abstract

Belief propagation is a remarkably effective tool for inference, even when applied to networks with cycles. It may be viewed as a way to seek the minimum of the Bethe free energy, though with no convergence guarantee in general. A variational perspective shows that, compared to exact inference, this minimization employs two forms of approximation: (i) the true entropy is approximated by the Bethe entropy, and (ii) the minimization is performed over a relaxation of the marginal polytope termed the local polytope. Here we explore when and how the Bethe approximation can fail for binary pairwise models by examining each aspect of the approximation, deriving results both analytically and with new experimental methods.

1 INTRODUCTION

Graphical models are a central tool in machine learning. However, the task of inferring the marginal distribution of a subset of variables, termed *marginal inference*, is NP-hard (Cooper, 1990), even to approximate (Dagum and Luby, 1993), and the closely related problem of computing the normalizing partition function is #P-hard (Valiant, 1979). Hence, much work has focused on finding efficient approximate methods. The sum-product message-passing algorithm termed belief propagation is guaranteed to return exact solutions if the underlying topology is a tree. Further, when applied to models with cycles, known as loopy belief propagation (LBP), the method is popular and often strikingly accurate (McEliece et al., 1998; Murphy et al., 1999).

A variational perspective shows that the true partition function and marginal distributions may be obtained by minimizing the true free energy over the marginal polytope. The standard Bethe approximation instead minimizes the Bethe free energy, which incorporates the Bethe pairwise approximation to the true entropy, over a relaxed pseudo-marginal

set termed the local polytope. A fascinating link to LBP was shown (Yedidia et al., 2001), in that fixed points of LBP correspond to stationary points of the Bethe free energy \mathcal{F} . Further, stable fixed points of LBP correspond to minima of \mathcal{F} (Heskes, 2003). Werner (2010) demonstrated a further equivalence to stationary points of an alternate function on the space of homogeneous reparameterizations.

In general, LBP may converge only to a local optimum or not converge at all. Various sufficient conditions have been derived for the uniqueness of stationary points (Mooij and Kappen, 2007; Watanabe, 2011), though convergence is often still not guaranteed (Heskes, 2004). Convergent methods based on analyzing derivatives of the Bethe free energy (Welling and Teh, 2001) and double-loop techniques (Heskes et al., 2003) have been developed. Recently, algorithms have been devised that are guaranteed to return an approximately stationary point (Shin, 2012) or a point with value ϵ -close to the optimum (Weller and Jebara, 2013a).

However, there is still much to learn about when and why the Bethe approximation performs well or badly. We shall explore both aspects of the approximation in this paper. Interestingly, sometimes they have opposing effects such that together, the result is better than with just one (see §4 for an example). We shall examine minima of the Bethe free energy over three different polytopes: marginal, local and cycle (see §2 for definitions). For experiments, we explore two methods, dual decomposition and Frank-Wolfe, which may be of independent interest. To provide another benchmark and isolate the entropy component, we also examine the tree-reweighted (TRW) approximation (Wainwright et al., 2005). Sometimes we shall focus on models where all edges are *attractive*, that is neighboring variables are pulled toward the same value; in this case it is known that the Bethe approximation is a lower bound for the true partition function (Ruozzi, 2012).

Questions we shall address include:

- In attractive models, why does the Bethe approximation perform well for the partition function but, when local potentials are low and coupling high, poorly for

marginals?

- In models with both attractive and repulsive edges, for low couplings, the Bethe approximation performs much better than TRW, yet as coupling increases, this advantage disappears. Can this be repaired by tightening the relaxation of the marginal polytope?
- Does tightening the relaxation of the marginal polytope always improve the Bethe approximation? In particular, is this true for attractive models?

This paper is organized as follows. Notation and preliminary results are presented in §2. In §3-4 we derive instructive analytic results, first focusing on the simplest topology that is not a tree, i.e. a single cycle. Already we observe interesting effects from both the entropy and polytope approximations. For example, even for attractive models, the Bethe optimum may lie outside the marginal polytope and tightening the relaxation leads to a worse approximation to the partition function. In §5 we examine more densely connected topologies, demonstrating a dramatic phase transition in attractive models as a consequence of the entropy approximation that leads to poor singleton marginals. Experiments are described in §6, where we examine test cases. Conclusions are discussed in §7. Related work is discussed throughout the text. An Appendix with technical details and proofs is attached in the Supplement.

2 NOTATION AND PRELIMINARIES

Throughout this paper, we restrict attention to binary pairwise Markov random fields (MRFs). We consider a model with n variables $X_1, \dots, X_n \in \mathbb{B} = \{0, 1\}$ and graph topology $(\mathcal{V}, \mathcal{E})$; that is \mathcal{V} contains nodes $\{1, \dots, n\}$ where i corresponds to X_i , and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ contains an edge for each pairwise relationship. Let $x = (x_1, \dots, x_n)$ be a configuration of all the variables, and $\mathcal{N}(i)$ be the neighbors of i . Primarily we focus on models with no ‘hard’ constraints, i.e. $p(x) > 0 \forall x$, though many of our results extend to this case. We may reparameterize the potential functions (Wainwright and Jordan, 2008) and define the energy E such that $p(x) = \frac{e^{-E(x)}}{Z}$ with

$$E = - \sum_{i \in \mathcal{V}} \theta_i x_i - \sum_{(i,j) \in \mathcal{E}} \frac{W_{ij}}{2} [x_i x_j + (1 - x_i)(1 - x_j)]. \quad (1)$$

This form allows edge coupling weights W_{ij} to be varied independently of the singleton potentials θ_i . If $W_{ij} > 0$ then an edge is *attractive*, if $W_{ij} < 0$ then it is *repulsive*. If all edges are attractive, then the model is attractive. We write μ_{ij} for pairwise marginals and, collecting together the θ_i and W_{ij} potential terms into a vector θ , with a slight abuse of notation, sometimes write (1) as $E = -\theta \cdot \mu$.

2.1 FREE ENERGY, VARIATIONAL APPROACH

Given any joint probability distribution $q(x)$ over all variables, the (Gibbs) free energy is defined as $\mathcal{F}_G(q) = \mathbb{E}_q(E) - S(q)$, where $S(q)$ is the (Shannon) entropy of the distribution.

It is easily shown (Wainwright and Jordan, 2008) that $-\log Z(\theta) = \min_q \mathcal{F}_G$, with the optimum when $q = p(\theta)$, the true distribution. This optimization is to be performed over all valid probability distributions, that is over the *marginal polytope*. However, this problem is intractable due to the difficulty of both computing the exact entropy S , and characterizing the polytope (Deza and Laurent, 2009).

2.2 BETHE APPROXIMATION

The standard approach of minimizing the *Bethe free energy* \mathcal{F} makes two approximations:

1. The entropy S is approximated by the *Bethe entropy*

$$S_B(\mu) = \sum_{(i,j) \in \mathcal{E}} S_{ij}(\mu_{ij}) + \sum_{i \in \mathcal{V}} (1 - d_i) S_i(\mu_i), \quad (2)$$

where S_{ij} is the entropy of μ_{ij} , S_i is the entropy of the singleton distribution of X_i and $d_i = |\mathcal{N}(i)|$ is the degree of i ; and

2. The marginal polytope is relaxed to the *local polytope*, where we require only local (pairwise) consistency, that is we deal with a *pseudo-marginal* vector q , that may not be globally consistent, which consists of $\{q_i = q(X_i = 1) \forall i \in \mathcal{V}, \mu_{ij} = q(x_i, x_j) \forall (i, j) \in \mathcal{E}\}$ subject to the constraints $q_i = \sum_{j \in \mathcal{N}(i)} \mu_{ij}$, $q_j = \sum_{i \in \mathcal{N}(j)} \mu_{ij} \forall i, j \in \mathcal{V}$.

In general, the Bethe entropy S_B is not concave and hence, the Bethe free energy $\mathcal{F} = E - S_B$ is not convex.

The global optimum of the Bethe free energy $\mathcal{F} = \mathbb{E}_q(E) - S_B(q)$ is achieved by minimizing \mathcal{F} over the local polytope, with the *Bethe partition function* Z_B defined such that the global minimum obtained equals $-\log Z_B$.

The local polytope constraints imply that, given q_i and q_j ,

$$\mu_{ij} = \begin{pmatrix} 1 + \xi_{ij} - q_i - q_j & q_j - \xi_{ij} \\ q_i - \xi_{ij} & \xi_{ij} \end{pmatrix} \quad (3)$$

for some $\xi_{ij} \in [0, \min(q_i, q_j)]$, where $\mu_{ij}(a, b) = q(X_i = a, X_j = b)$.

As in (Welling and Teh, 2001), one can solve for the Bethe optimal ξ_{ij} explicitly in terms of q_i and q_j by minimizing \mathcal{F} , leading to

$$\xi_{ij}^*(q_i, q_j) = \frac{1}{2\alpha_{ij}} \left(Q_{ij} - \sqrt{Q_{ij}^2 - 4\alpha_{ij}(1 + \alpha_{ij})q_i q_j} \right), \quad (4)$$

where $\alpha_{ij} = e^{W_{ij}} - 1$, $Q_{ij} = 1 + \alpha_{ij}(q_i + q_j)$.

Thus, we may consider the Bethe approximation as minimizing \mathcal{F} over $q = (q_1, \dots, q_n) \in [0, 1]^n$. Further, the derivatives are given by

$$\frac{\partial \mathcal{F}}{\partial q_i} = -\phi_i + \log \left[\frac{(1 - q_i)^{d_i - 1}}{q_i^{d_i - 1}} \prod_{j \in N(i)} \frac{(q_i - \xi_{ij}^*)}{(1 + \xi_{ij}^* - q_i - q_j)} \right], \quad (5)$$

where $\phi_i = \theta_i - \frac{1}{2} \sum_{j \in N(i)} W_{ij}$.

2.3 TREE-REWEIGHTED APPROXIMATION

Our primary focus in this paper is on the Bethe approximation but we shall find it helpful to compare results to another form of approximate inference. The *tree-reweighted* (TRW) approach may be regarded as a family of variational methods, where first one selects a point from the *spanning tree polytope*, that is the convex hull of all spanning trees of the model, represented as a weighting for each edge. Given this selection, the corresponding TRW entropy is the weighted combination of entropies on each of the possible trees. This is then combined with the energy and optimized over the local polytope, similarly to the Bethe approximation. Hence it provides an interesting contrast to the Bethe method, allowing us to focus on the difference in the entropy approximation. An important feature of TRW is that its entropy is concave and always upper bounds the true entropy (neither property is true in general for the Bethe entropy). Hence minimizing the TRW free energy is a convex problem and yields an upper bound on the true partition function. Sometimes we shall consider the optimal upper bound, i.e. the lowest upper bound achievable over all possible selections from the spanning tree polytope.

2.4 CYCLE POLYTOPE

We shall consider an additional relaxation of the marginal polytope termed the *cycle polytope*. This inherits all constraints of the local polytope, hence is at least as tight, and in addition enforces consistency around any cycle. A polyhedral approach characterizes this by requiring the following *cycle inequalities* to be satisfied (Barahona, 1993; Deza and Laurent, 2009; Sontag, 2010) for all cycles C and every subset of edges $F \subseteq C$ with $|F|$ odd:

$$\sum_{(i,j) \in F} (\mu_{ij}(0,0) + \mu_{ij}(1,1)) + \sum_{(i,j) \in C \setminus F} (\mu_{ij}(1,0) + \mu_{ij}(0,1)) \geq 1. \quad (6)$$

Each cycle inequality describes a facet of the marginal polytope (Barahona and Mahjoub, 1986). It is typically easier to optimize over the cycle polytope than the marginal polytope, and earlier work has shown that results are often similar (Sontag and Jaakkola, 2007).

2.5 SYMMETRIC AND HOMOGENEOUS MRFS

For analytic tractability, we shall often focus on particular forms of MRFS. We say a MRF is *homogeneous* if all singleton potentials are equal, all edge potentials are equal, and its graph has just one vertex and edge orbit.¹

A MRF is *symmetric* if it has no singleton potentials, hence flipping all variables $0 \leftrightarrow 1$ leaves the energy unchanged, and the true marginals for each variable are $(\frac{1}{2}, \frac{1}{2})$. For symmetric, planar binary pairwise MRFS, it is known that the cycle polytope is equal to the marginal polytope (Barahona and Mahjoub, 1986). Using (4) and (5), it is easy to show the following result.

Lemma 1. *The Bethe free energy of any symmetric MRF has a stationary point at $q_i = \frac{1}{2} \forall i$.*

We remark that this is *not* always a minimum (see §5).

2.6 DERIVATIVES AND MARGINALS

It is known that the derivatives of $\log Z$ with respect to the potentials are the marginals, and that this also holds for any convex free energy, where pseudo-marginals replace marginals if a polytope other than the marginal is used (Wainwright, 2006). Using Danskin's theorem (Bertsekas, 1995), this can be generalized as follows.

Lemma 2. *Let $\hat{F} = E - \hat{S}(\mu)$ be any free energy approximation, X be a compact space, and $\hat{A} = -\min_{\mu \in X} \hat{F}$ be the corresponding approximation to $\log Z$.*

If the arg min is unique at pseudo-marginals τ ,

then $\frac{\partial \hat{A}}{\partial \theta_i} = \tau_i(1)$, $\frac{\partial \hat{A}}{\partial W_{ij}} = \tau_{ij}(0,0) + \tau_{ij}(1,1)$.

If the arg min is not unique then let $Q(\theta)$ be the set of arg mins; the directional derivative of \hat{A} in direction $\theta \leftarrow \theta + y$ is given by $\nabla_y \hat{A} = \max_{\tau \in Q(\theta)} \tau \cdot y$.

In the next Section we begin to apply these results to analyze the locations and values of the minima of the Bethe free energy.

3 HOMOGENEOUS CYCLES

Since the Bethe approximation is exact for models with no cycles, it is instructive first to consider the case of one cycle on n variables, which we write as C_n . Earlier analysis considered the perspective of belief updates (Weiss, 2000; Aji, 2000). Here we examine the Bethe free energy, which in this context is convex (Pakzad and Anantharam, 2002) with a unique optimum.² We consider symmetric models, initially analyzing the homogeneous case.

¹This means there is a graph isomorphism mapping any edge to any other, and the same for any vertex.

²This follows by considering (2) and observing that $S_{ij} - S_i$ (conditional entropy) is concave over the local consistency constraints, hence by appropriate counting, the total Bethe entropy is concave provided an MRF has at most one cycle.

With Lemma 1, we see that singleton marginals are $\frac{1}{2}$ across all approximation methods. For pairwise marginals, the following result holds due to convexity.

Lemma 3. *For any symmetric MRF and a free energy that is convex, the optimum occurs at uniform pseudo-marginals across all pairs of variables, either where the derivative is zero or at an extreme point of the range.*

The uniformity of the optimal edge pseudo-marginals, together with Lemma 1, shows that all are $\mu_{ij} = \begin{pmatrix} x & \frac{1}{2} - x \\ \frac{1}{2} - x & x \end{pmatrix} \forall (i, j) \in \mathcal{E}$, where just x remains to be identified. The optimum x with zero derivative is always contained within the local polytope but we shall see that this is not always the case when we consider the cycle relaxation. Using (4), it is straightforward to derive the following result for the Bethe pairwise marginals.

Lemma 4. *For a symmetric homogeneous cycle, the Bethe optimum over the local polytope is at $x = x_B(W) = \frac{1}{2} \sigma(W/2)$, where we use standard sigmoid $\sigma(y) := \frac{1}{1+e^{-y}}$. Observe that $x_B(-W) = 1/2 - x_B(W)$.*

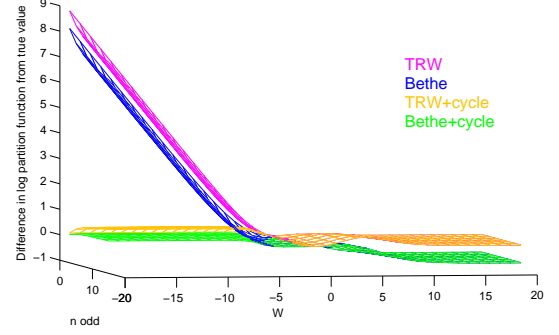
Further, we can derive the error of the Bethe pairwise marginals by using the loop series result given in Lemma 5 of §4, taking log, differentiating and using Lemma 2, to give the difference between true x and Bethe x_B as

$$x - x_B = \frac{1}{4} \frac{\text{sech}^2 \frac{W}{4} \tanh^{n-1} \frac{W}{4}}{1 + \tanh^n \frac{W}{4}}. \quad (7)$$

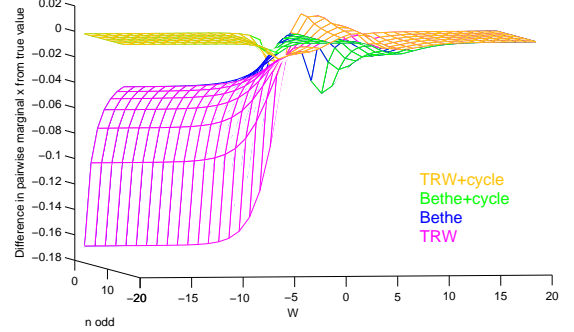
Remarks: Observe that at $W = 0$, $x - x_B = 0$; as $W \rightarrow \pm\infty$, $x - x_B \rightarrow 0$. For $W \neq 0$, $x - x_B$ is always > 0 unless n is even and $W < 0$, in which case it is negative. Differentiating (7) and solving for where x and x_B are most apart gives empirically $W \approx 2 \log n + 0.9$ with corresponding max value of $x - x_B \approx \frac{1}{5n}$ for large n .

See Figure 1 for plots, where, for TRW, values were computed using optimal edge weights, as derived in the Appendix. Observe that at $W = 0$, all methods are exact. As W increases, the Bethe approximations to both $\log Z$ and the marginal x rise more slowly than the true values, though once W is high enough that x is large and cannot rise much further, then the Bethe x_B begins to catch up until they are both close to $\frac{1}{2}$ for large W . We remark that since the Bethe approximation is always a lower bound on the partition function for an attractive model (Ruozzi, 2012), and both the partition functions and marginals are equal at $W = 0$, we know from Lemma 2 that x_B must rise more slowly than x , as seen.

For $W > 0$, tightening the polytope makes no difference. The picture is different for negative W if n is odd, in which case we have a *frustrated cycle*, that is a cycle with an odd number of repulsive edges, which often causes difficulties with inference methods (Weller and Jebara, 2013b).



(a) Errors of $\log Z$ approximations



(b) Errors of pairwise marginal x

Figure 1: Homogeneous cycle C_n , n odd, edge weights W . By Lemma 2, the slope of the error of $\log Z$ wrt W is twice the error of x . For $W > 0$, local and cycle polytopes have the same values.

In this case, (6) is binding for $W < -2 \log(n-1)$ and prevents the Bethe+cycle marginal x_{BC} from falling below $\frac{1}{2n}$. As $W \rightarrow -\infty$, the true x also does not fall below $\frac{1}{2n}$.³ Thus, as $W \rightarrow -\infty$, the score (negative energy) and hence $\log Z \rightarrow -\infty$ for the true distribution. This also holds for Bethe or TRW on the cycle polytope, but on the local polytope, their energy and $\log Z \rightarrow 0$. Observe that for $W < 0$, Bethe generally outperforms TRW over both polytopes.

Tables 1 and 2 summarize results as $W \rightarrow \pm\infty$, again using optimal edge weights for TRW.

Model	$W \rightarrow -\infty$		$W \rightarrow \infty$	
	$\log Z'$	x	$\log \frac{Z'}{Z}$	x
Bethe	0	0	$-\log 2$	$1/2$
Bethe+cycle	0	0	$-\log 2$	$1/2$
TRW	$\log 2$	0	0	$1/2$
TRW+cycle	$\log 2$	0	0	$1/2$
True distribution	$\log 2$	0	0	$1/2$

Table 1: Analytic results for homogenous cycle C_n , n even. As $W \rightarrow \infty$, $\log Z'$ and $\log Z \rightarrow \infty$ so the difference is shown.

³To see this, note there are $2n$ configurations whose probabilities dominate as $W \rightarrow -\infty$: $01 \dots 0$, its inverse flipping $0 \leftrightarrow 1$, and all n rotations; of these, just one has 00 and one has 11 for a specific edge.

Model	$W \rightarrow -\infty$		$W \rightarrow \infty$	
	$\log Z'$	x	$\log \frac{Z'}{Z}$	x
Bethe	0	0	$-\log 2$	1/2
Bethe+cycle	$-\infty$	$1/(2n)$	$-\log 2$	1/2
TRW	$\log 2$	0	0	1/2
TRW+cycle	$-\infty$	$1/(2n)$	0	1/2
True distribution	$-\infty$	$1/(2n)$	0	1/2

Table 2: Analytic results for homogeneous cycle C_n, n odd. As $W \rightarrow \infty$, $\log Z'$ and $\log Z \rightarrow \infty$ so the difference is shown.

4 NONHOMOGENEOUS CYCLES

The loop series method (Chertkov and Chernyak, 2006; Sudderth et al., 2007) provides a powerful tool to analyze the ratio of the true partition function to its Bethe approximation. In symmetric models with at most one cycle, by Lemma 3, we know that the unique Bethe optimum is at uniform marginals $q_i = \frac{1}{2}$. Using this and (4), and substituting into the loop series result yields the following.

Lemma 5. *For a symmetric MRF which includes exactly one cycle C_n , with edge weights W_1, \dots, W_n , then $Z/Z_B = 1 + \prod_{i=1}^n \tanh \frac{W_i}{4}$.*

Remarks: In this setting, the ratio Z/Z_B is always ≤ 2 and ≈ 1 if even one cycle edge is weak, as might be expected since then the model is almost a tree. The ratio has no dependence on edges not in the cycle and those pairwise marginals will be exact. Further, since the Bethe entropy is concave, by Lemma 1, all singleton marginals are exact at $\frac{1}{2}$. Errors of pairwise pseudo-marginals on the cycle can be derived by using the expression for Z/Z_B from Lemma 5, taking log then differentiating and using Lemma 2.

Several principles are illustrated by considering 3 variables, A, B and C , connected in a triangle. Suppose AB and AC have strongly attractive edges with weight $W = 10$. We examine the effect of varying the weight of the third edge BC , see Figure 2.

It was recently proved (Ruozi, 2012) that $Z_B \leq Z$ for attractive models. A natural conjecture is that the Bethe optimum pseudo-marginal in the local polytope must lie inside the marginal polytope. However, our example, when BC is weakly attractive, proves this conjecture to be false. As a consequence, tightening the local polytope to the marginal polytope for the Bethe free energy in this case worsens the approximation of the log-partition function (though it improves the marginals), see Figure 2 near 0 BC edge weight. For this model, the two aspects of the Bethe approximation to $\log Z$ act in opposing directions - the result is more accurate with both than with either one alone. For intuition, note that via the path $B-A-C$, in the globally consistent probability distribution, B and C are overwhelmingly likely to take the same value. Given that singleton marginals are $\frac{1}{2}$, the Bethe approximation, however, decomposes into a sep-

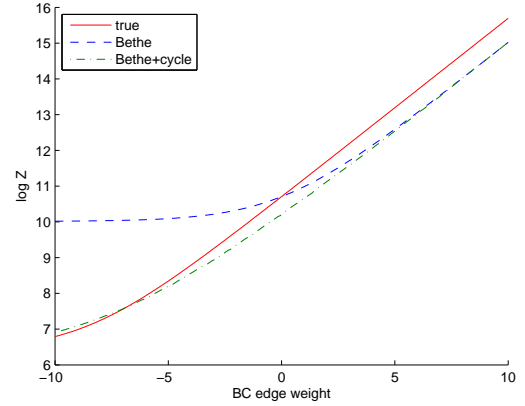


Figure 2: Log partition function and approximations for ABC triangle, see §4. Edge weights for AB and AC are 10 (strongly attractive) while BC is varied as shown. Near 0: Bethe is a better approximation to $\log Z$ but Bethe+cycle has better derivative, hence better marginals by Lemma 2; since Bethe+cycle is below Bethe in this region, its optimum does not lie in the local polytope.

arate optimization for each edge, which for the weak edge BC , yields that B and C are almost independent, leading to a conflict with the true marginal. This causes the Bethe optimum over the local polytope to lie outside the marginal polytope. The same conclusion may be drawn rigorously by considering the cycle inequality (6), taking the edge set $F = \{BC\}$ and observing that the terms are approximately $\frac{1}{4} + \frac{1}{4} + 2(0 + 0) \approx \frac{1}{2} < 1$. Recall that here the cycle and marginal polytopes are the same (see §2.5). The same phenomenon can also be shown to occur for the TRW approximation with uniform edge appearance probabilities.

Notice in Figure 2 that as the BC edge strength rises above 0, the Bethe marginals (given by the derivative) improve while the $\log Z$ approximation deteriorates. We remark that the exactness of the Bethe approximation on a tree can be very fragile in the sense that adding a very weak edge between variables to complete a cycle may expose that pairwise marginal as being (perhaps highly) inaccurate.

5 GENERAL HOMOGENEOUS GRAPHS

We discuss how the Bethe entropy approximation leads to a ‘phase shift’ in behavior for graphs with more than one cycle when W is above a positive threshold.

The true entropy is always maximized at $q_i = \frac{1}{2}$ for all variables. This also holds for the TRW approximation. However, in densely connected attractive models, the Bethe approximation pulls singleton marginals towards 0 or 1. This behavior has been discussed previously (Heskes, 2004; Mooij and Kappen, 2005) and described in terms of algorithmic stability (Wainwright and Jordan, 2008, §7.4), or heuristically as a result of LBP over-counting information when going around cycles (Ihler, 2007), but here we

explain it as a consequence of the Bethe entropy approximation.

We focus on symmetric homogeneous models which are d -regular, i.e. each node has the same degree d . One example is the complete graph on n variables, K_n . For this model, $d = n - 1$. The following result is proved in the Appendix, using properties of the Hessian from (Weller and Jebara, 2013a).

Lemma 6. *Consider a symmetric homogeneous MRF on n vertices with d -regular topology and edge weights W . $q = (\frac{1}{2}, \dots, \frac{1}{2})$ is a stationary point of the Bethe free energy but for W above a critical value, this is not a minimum. Specifically, let H be the Hessian of the Bethe free energy at q , x_B be the value from Lemma 4 and $\mathbf{1}$ be the vector of length n with 1 in each dimension; then $\mathbf{1}^T H \mathbf{1} = n[d - 4x_B(d - 1)]/x_B < 0$ if $x_B > \frac{1}{4} \frac{d}{d-1} \Leftrightarrow W > 2 \log \frac{d}{d-2}$.*

To help understand this result, consider (2) for the Bethe entropy S_B , and recall that $\sum_i d_i = 2m$ (m is the number of edges, handshake lemma), hence $S_B = mS_{ij} - (2m - n)S_i$. For large W , all the probability mass for each edge is pulled onto the main diagonal, thus $S_{ij} \approx S_i$. For $m > n$, which interestingly is exactly the case of more than one cycle, in order to achieve the optimum S_B , each entropy term $\rightarrow 0$ by tending to pairwise marginal $\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$ or symmetrically $\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$. See the second row of Figure 3 for an illustration of how the Bethe entropy surface changes dramatically as W rises, even sometimes going negative, and the top row to see how the Bethe free energy surfaces changes rapidly as W moves through the critical threshold.

Reinforcing this pull of singleton marginals away from $\frac{1}{2}$ is the shape of the energy surface, when optimized for free energy subject to given singleton marginals. In the Bethe approximation, this is achieved by computing ξ_{ij} terms according to (4), as illustrated in the bottom row of Figure 3, but for any reasonable entropy term (including TRW), always $\xi_{ij} < \min(q_i, q_j)$, hence the energy is lower towards the extreme values 0 or 1.

Remarks: (i) This effect is specifically due to the Bethe entropy approximation, and is not affected by tightening the polytope relaxation, as we shall see in §6. (ii) To help appreciate the consequences of Lemma 6, observe that $\log \frac{d}{d-2}$ is positive, monotonically decreasing to 0 as d increases. Thus, for larger, more densely connected topologies, the threshold for this effect is at lower positive edge weights. Above the threshold, $q_i = \frac{1}{2}$ is no longer a minimum but becomes a saddle point.⁴ (iii) This explains the observation made after (Heinemann and Globerson, 2011,

⁴The Hessian at $q_i = \frac{1}{2}$ is neither positive nor negative definite. Moving away from the valley where all q_i are equal, the Bethe free energy rises quickly.

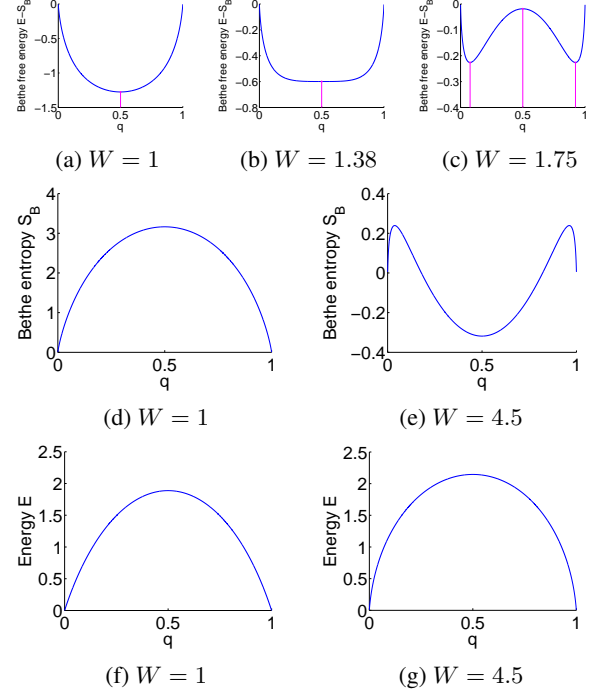


Figure 3: Bethe free energy $E - S_B$ with stationary points highlighted (top), then entropy S_B (middle) and energy E (bottom) vs $q_i = q \forall i$ for symmetric homogeneous complete graph K_5 . **All quantities are evaluated at the optimum over pairwise marginals**, i.e. $\{\xi_{ij}\}$ are computed as in (4). These figures are described in Lemma 6 and the text thereafter. $W \approx 1.38$ is the critical threshold, above which Bethe singleton marginals are rapidly pulled toward 0 or 1. $W = 4.5$ is sufficiently high that the Bethe entropy becomes negative at $q = \frac{1}{2}$ (middle row).

Lemma 3), where it is pointed out that for an attractive model as $n \rightarrow \infty$, if $n/m \rightarrow 0$, a marginal distribution (other than the extreme of all 0 or all 1) is unlearnable by the Bethe approximation (because the effect we have described pushes all singleton marginals to 0 or 1). (iv) As W rises, although the Bethe singleton marginals can be poor, the Bethe partition function does not perform badly: For a symmetric model, as $W \rightarrow \infty$, there are 2 dominating MAP states (all 0 or all 1) with equal probability. The true marginals are at $q_i = \frac{1}{2}$ which picks up the benefit of $\log 2$ entropy, whereas the Bethe approximation converges to one or other of the MAP states with 0 entropy, hence has $\log 2$ error.

To see why a similar effect does not occur as $W \rightarrow -\infty$, note that for $W < 0$ around a frustrated cycle, the minimum energy solution on the local polytope is at $q_i = \frac{1}{2}$. Indeed, this can pull singleton Bethe marginals toward $\frac{1}{2}$ in this case. See §5.1 in the Appendix for further analysis.

6 EXPERIMENTS

We are interested in the empirical performance of the optimum Bethe marginals and partition function, as the re-

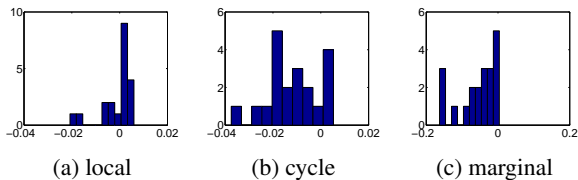


Figure 4: Histogram of differences observed in optimum returned Bethe free energy, FW-mesh primal, over the 20 models in the validation set (mesh using $\epsilon = 0.1$, less than ϵ is insignificant). Negative numbers indicate FW outperformed mesh.

laxation of the marginal polytope is tightened. Many methods have been developed to attempt the optimization over the local polytope, primarily addressing its non-convexity, though none is guaranteed to return the global optimum. Recently, an algorithm was derived to return an ϵ -approximation to the optimum $\log Z_B$ based on constructing a discretized mesh of pseudo-marginals (Weller and Jebara, 2013a, 2014). One method for optimizing over tighter relaxations is to use this algorithm as an inner solver in an iterative dual decomposition approach with subgradient updates (Sontag, 2010; Sontag et al., 2011), where it can be shown that, when minimizing the Bethe free energy, the dual returned less ϵ lower bounds $-\log Z_B$ over the tighter polytope. This would be our preferred approach, but for the models on which we would like to run experiments, the runtime is prohibitive.

Hence we explored two other methods: (i) We replaced the inner solver with a faster, convergent double-loop method, the HAK-BETHE option in libDAI (Heskes et al., 2003; Mooij, 2010), though this is guaranteed only to return a local optimum at each iteration, hence we have no guarantee on the quality of the final result; (ii) We applied the Frank-Wolfe algorithm (FW) (Frank and Wolfe, 1956; Jaggi, 2013; Belanger et al., 2013). At each iteration, a tangent hyperplane is computed at the current point, then a move is made to the best computed point along the line to the vertex (of the appropriate polytope) with the optimum score on the hyperplane. This proceeds monotonically, even on a non-convex surface such as the Bethe free energy, hence will converge (since it is bounded), though runtime is guaranteed only for a convex surface as in TRW.

FW can be applied directly to optimize over marginal, cycle or local polytopes, and performed much better than HAK: runtime was orders of magnitude faster, and the energy found was in line with HAK.⁵ To further justify using FW, which may only reach a local optimum, on our main test cases, we compared its performance on a small validation set against the benchmark of dual decomposition using the guaranteed ϵ -approximate mesh method (Weller and Jebara, 2014) as an inner solver.

⁵The average difference between energies found was < 0.1 .

6.1 IMPLEMENTATION AND VALIDATION

To validate FW for the Bethe approximations on each polytope, we compared log partition functions and pairwise marginals across 20 MRFs, each on a complete graph with 5 variables. Each edge potential was drawn $W_{ij} \sim [-8, 8]$ and each singleton potential $\theta_i \sim [-2, 2]$. To handle the tighter polytope relaxations using the mesh method, we used a dual decomposition approach as follows. For the cycle polytope, one Lagrangian variable was introduced for each cycle constraint (6) with projected subgradient descent updates. For the marginal polytope, rather than imposing each facet constraint, which would quickly become unmanageable⁶, instead a lift-and-project method was employed (Sontag, 2010). These algorithms may be of independent interest and are provided in the Supplement.

For all mesh runs, we used $\epsilon = 0.1$. Note that strong duality is not guaranteed for Bethe since the objective is non-convex, hence we are guaranteed only an upper bound on $\log Z_B$; yet we were able to monitor the duality gap by using rounded primals and observed that the realized gaps were typically within ϵ , see Figure 6.

For FW, we always initialized at the uniform distribution, i.e. $\mu_{ij} = \left(\frac{1}{4}, \frac{1}{4}\right) \forall (i, j) \in \mathcal{E}$, note this is always within the marginal polytope. At each iteration, to determine how far to go along the line to the optimum vertex, we used Matlab’s `fminbnd` function. This induces a minimum move of 10^{-6} along the line to the optimum vertex, which was helpful in escaping from local minima. When we tried allowing zero step size, performance became worse. Our stopping criterion was to run for 10,000 iterations (which did not take long) or until the objective value changed by $< 10^{-6}$, at which point we output the best value found so far, and the corresponding pseudo-marginals.

Results on the validation set are shown in Figure 4, indicating that FW performed well compared to mesh + dual decomposition (the best standard we have for the Bethe optimum). Note, however, that good performance on $\log Z_B$ estimation does not necessarily imply that the Bethe optimal marginals were being returned for either method. There may be several local optima where the Bethe free energy has value close to the global optimum, and methods may return different locations. This is a feature of the non-convex surface which should be borne in mind when considering later results, hence we should not be surprised that in the validation set, although 17/20 of the runs had ℓ_1 error in singleton marginals under 0.05, there were 3 runs with larger differences, in one case as high as 0.7 (not shown).⁷

⁶The number of facets of the marginal polytope grows extremely rapidly (Deza and Laurent, 2009).

⁷Recall the example from §5, where a symmetric homogeneous MRF with complete graph K_n topology and high edge

Given this performance, we used FW for all Bethe optimizations on the test cases. FW was also used for all TRW runs, where edge appearance probabilities were obtained using the matrix-tree theorem with weights proportional to each edge’s coupling strength $|W_{ij}|$, as was used in (Sontag and Jaakkola, 2007).

6.2 TEST SETS

Models with 10 variables connected in a complete graph were drawn with random potentials. This allows comparison to earlier work such as (Sontag and Jaakkola, 2007) and (Meshi et al., 2009, Appendix). In addition to examining error in log partition functions and singleton marginals as was done in earlier work, given our theoretical observations in §3-5, we also explored the error in pairwise marginals. To do this, we report the ℓ_1 error in the estimated probability that a pair of variables is equal, averaged over all edges, i.e. we report average ℓ_1 error of $\mu_{ij}(0, 0) + \mu_{ij}(1, 1)$. We used FW to minimize the Bethe and TRW free energies over each of the local, cycle and marginal polytopes. For each maximum coupling value used, 100 models were generated and results averaged for plotting. Given the theoretical observations of §3-5, we are interested in behavior both for attractive and general (non-attractive) models.

For general models, potentials were drawn for single variables $\theta_i \sim U[-2, 2]$ and edges $W_{ij} \sim U[-y, y]$ where y was varied to observe the impact of coupling strength.⁸ Results are shown in Figure 5. Tightening the relaxation of the polytope from local to cycle or marginal, dramatically improves both Bethe and TRW approximations on all measures, with little difference between the cycle or marginal polytopes. This confirms observations in (Sontag and Jaakkola, 2007).

The relative performance of Bethe compared to TRW depends on the criteria used. Looking at the error of singleton marginals, Bethe is better than TRW for low coupling strengths, but for high coupling strengths the methods perform equally well on the local polytope, whereas on the cycle or marginal polytopes, TRW outperforms Bethe (though Bethe is still competitive). Thus, tightening the relaxation of the local polytope at high coupling does not lead to Bethe being superior on all measures. However, in terms of partition function and pairwise marginals, which are important in many applications, Bethe does consistently outperform TRW in all settings, and over all polytopes.

For attractive models, in order to explore our observations in §5, much lower singleton potentials were used. We drew

weights was shown to have 2 locations at the global minimum, with average ℓ_1 distance between them approaching 1.

⁸These settings were chosen to facilitate comparison with the results of (Sontag and Jaakkola, 2007), though in that paper, variables take values in $\{-1, 1\}$ so the equivalent singleton potential ranges coincide. To compare couplings, our y values should be divided by 4.

$\theta_i \sim U[-0.1, 0.1]$ and $W_{ij} \sim U[0, y]$ where y is varied. This is consistent with parameters used by Meshi et al. (2009). Results are shown in Figure 7. When coupling is high, the Bethe entropy approximation pushes singleton marginals away from $\frac{1}{2}$. This effect quickly becomes strong above a threshold. Hence, when singleton potentials are very low, i.e. true marginals are close to $\frac{1}{2}$, the Bethe approximation will perform poorly irrespective of polytope, as observed in our attractive experiments. We note, however, that this effect rarely causes singleton marginals to cross over to the other side of $\frac{1}{2}$. Further, as discussed in §5, the partition function approximation is not observed to deviate by more than $\log 2$ on average.

7 CONCLUSIONS

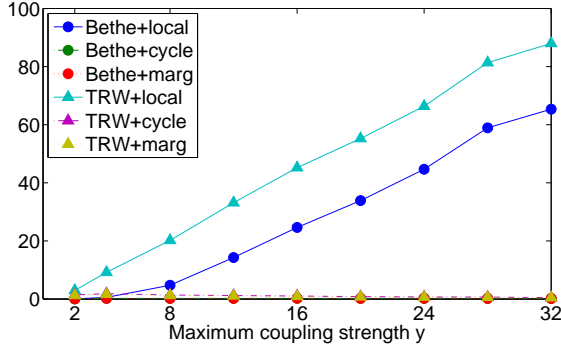
We have used analytic and empirical methods to explore the two aspects of the Bethe approximation: the polytope relaxation and the entropy approximation. We found Frank-Wolfe to be an effective method for optimization, and note that for the cycle polytope, the runtime of each iteration scales polynomially with the number of variables (see §6.1.3 in the Appendix for further details).

For general models with both attractive and repulsive edges, tightening the relaxation of the polytope from local to cycle or marginal, dramatically improves both Bethe and TRW approximations on all measures, with little difference between the cycle or marginal polytopes. For singleton marginals, except when coupling is low, there does not appear to be a significant advantage to solving the non-convex Bethe free energy formulation compared to convex variational approaches such as TRW. However, for log-partition function estimation, Bethe does provide significant benefits. Empirically, in both attractive and mixed models, Bethe pairwise marginals appear consistently better than TRW.

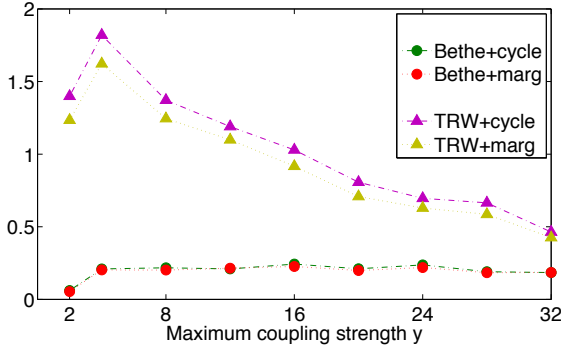
In our experiments with attractive models, the polytope approximation appears to make little difference. However, we have shown theoretically that in some cases it can cause a significant effect. In particular, our discussion of non-homogeneous attractive cycles in §4 shows that even in the attractive setting, tightening the polytope can affect the Bethe approximation - improving marginals but worsening the partition function. It is possible that to observe this phenomenon empirically, one needs a different distribution over models.

Acknowledgements

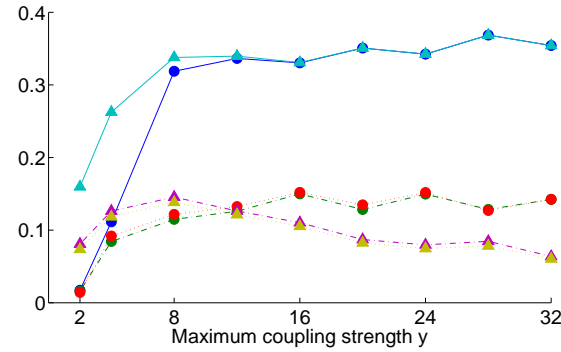
We thank U. Heinemann and A. Globerson for helpful correspondence. Work by A.W., K. T. and T.J. was supported in part by NSF grants IIS-1117631 and CCF-1302269. Work by D.S. was supported in part by the DARPA PPAML program under AFRL contract no. FA8750-14-C-0005.



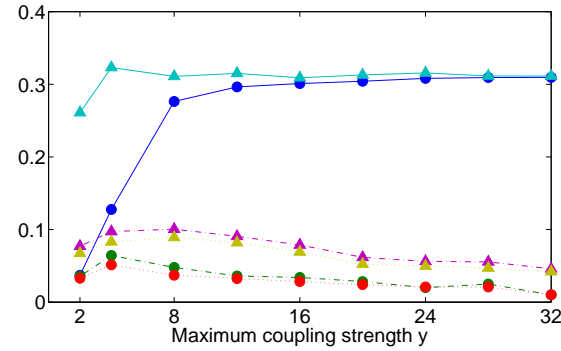
(a) log partition error



(b) log partition error, local polytope removed



(c) Singleton marginals, average ℓ_1 error



(d) Pairwise marginals, average ℓ_1 error

Figure 5: Results for general models showing error vs true values. $\theta_i \sim \mathcal{U}[-2, 2]$. **The legend is consistent across plots.** These may be compared to plots in (Sontag and Jaakkola, 2007).

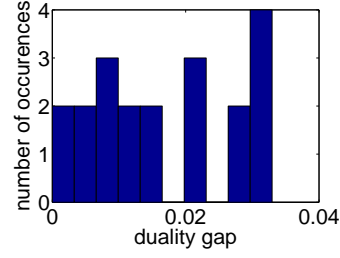
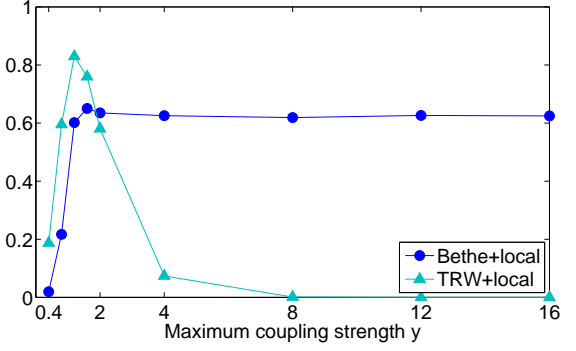
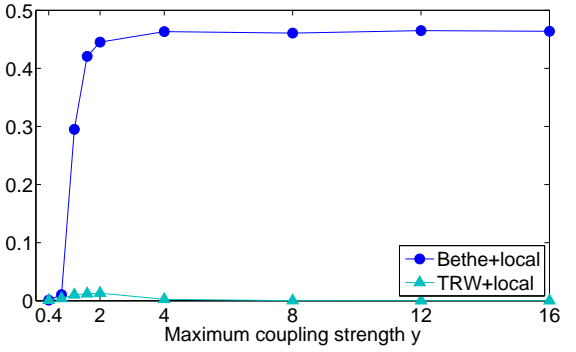


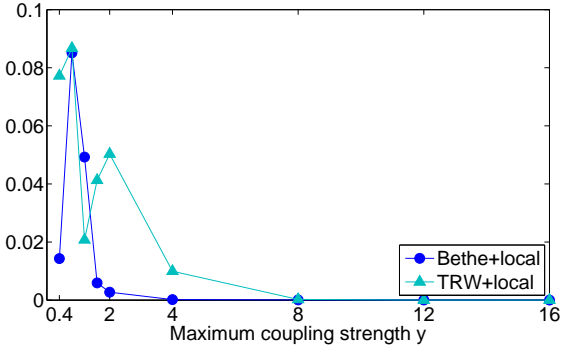
Figure 6: Duality gaps observed on the validation set using mesh approach + dual decomposition over 20 models, cycle polytope, $\epsilon = 0.1$. See text in §6.1



(a) log partition error



(b) Singleton marginals, average ℓ_1 error



(c) Pairwise marginals, average ℓ_1 error. **Note small scale.**

Figure 7: Results for attractive models showing error vs true values. $\theta_i \sim \mathcal{U}[-0.1, 0.1]$. Only local polytope shown, **results for other polytopes are almost identical.**

References

- S. Aji. *Graphical models and iterative decoding*. PhD thesis, California Institute of Technology, 2000.
- F. Barahona. On cuts and matchings in planar graphs. *Math. Program.*, 60:53–68, 1993.
- F. Barahona and A. Mahjoub. On the cut polytope. *Mathematical Programming*, 36(2):157–173, 1986. ISSN 0025-5610. doi: 10.1007/BF02592023.
- D. Belanger, D. Sheldon, and A. McCallum. Marginal inference in MRFs using Frank-Wolfe. In *NIPS Workshop on Greedy Optimization, Frank-Wolfe and Friends*, December 2013.
- D. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1995.
- S. Boyd and A. Mutapcic. Subgradient Methods, notes for EE364b, Jan 2007. http://www.stanford.edu/class/ee364b/notes/subgrad_method_notes.pdf, 2007.
- M. Chertkov and M. Chernyak. Loop series for discrete statistical models on graphs. *J. Stat. Mech.*, 2006.
- G. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42:393–405, 1990.
- P. Dagum and M. Luby. Approximate probabilistic reasoning in Bayesian belief networks is NP-hard. *Artificial Intelligence*, 60:141–153, 1993.
- M. Deza and M. Laurent. *Geometry of Cuts and Metrics*. Springer Publishing Company, Incorporated, 1st edition, 2009. ISBN 3642042945, 9783642042942.
- M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, 1956. ISSN 1931-9193. doi: 10.1002/nav.3800030109.
- U. Heinemann and A. Globerson. What cannot be learned with Bethe approximations. In *UAI*, pages 319–326, 2011.
- T. Heskes. Stable fixed points of loopy belief propagation are minima of the Bethe free energy. In *Neural Information Processing Systems*, 2003.
- T. Heskes. On the uniqueness of loopy belief propagation fixed points. *Neural Computation*, 16(11):2379–2413, 2004.
- T. Heskes, K. Albers, and B. Kappen. Approximate inference and constrained optimization. In *UAI*, pages 313–320, 2003.
- A. Ihler. Accuracy bounds for belief propagation. In *Uncertainty in Artificial Intelligence (UAI)*, 2007.
- M. Jaggi. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In *ICML (1)*, pages 427–435, 2013.
- R. McEliece, D. MacKay, and J. Cheng. Turbo decoding as an instance of Pearl’s “Belief Propagation” algorithm. *IEEE Journal on Selected Areas in Communications*, 16(2):140–152, 1998.
- O. Meshi, A. Jaimovich, A. Globerson, and N. Friedman. Convexifying the Bethe free energy. In *UAI*, pages 402–410, 2009.
- J. Mooij. libDAI: A free and open source C++ library for discrete approximate inference in graphical models. *Journal of Machine Learning Research*, 11:2169–2173, August 2010. URL <http://www.jmlr.org/papers/volume11/mooij10a/mooij10a.pdf>.
- J. Mooij and H. Kappen. On the properties of the Bethe approximation and loopy belief propagation on binary networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2005.
- J. Mooij and H. Kappen. Sufficient conditions for convergence of the sum-product algorithm. *IEEE Transactions on Information Theory*, 53(12):4422–4437, December 2007.
- K. Murphy, Y. Weiss, and M. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Uncertainty in Artificial Intelligence (UAI)*, 1999.
- P. Pakzad and V. Anantharam. Belief propagation and statistical physics. In *Princeton University*, 2002.
- N. Ruozzi. The Bethe partition function of log-supermodular graphical models. In *Neural Information Processing Systems*, 2012.
- J. Shin. Complexity of Bethe approximation. In *Artificial Intelligence and Statistics*, 2012.
- D. Sontag. *Approximate Inference in Graphical Models using LP Relaxations*. PhD thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 2010.
- D. Sontag and T. Jaakkola. New outer bounds on the marginal polytope. In *NIPS*, 2007.
- D. Sontag, A. Globerson, and T. Jaakkola. Introduction to dual decomposition for inference. In S. Sra, S. Nowozin, and S. Wright, editors, *Optimization for Machine Learning*. MIT Press, 2011.
- E. Sudderth, M. Wainwright, and A. Willsky. Loop series and Bethe variational bounds in attractive graphical models. In *NIPS*, 2007.
- L. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8(2):189–201, 1979.
- M. Wainwright. Estimating the “wrong” graphical model: Benefits in the computation-limited setting. *Journal of Machine Learning Research*, 7:1829–1859, 2006.
- M. Wainwright and M. Jordan. Graphical models, exponential families and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.
- M. Wainwright, T. Jaakkola, and A. Willsky. A new class of upper bounds on the log partition function. *IEEE Transactions on Information Theory*, 51(7):2313–2335, 2005.
- Y. Watanabe. Uniqueness of belief propagation on signed graphs. In *Neural Information Processing Systems*, 2011.
- Y. Weiss. Correctness of local probability propagation in graphical models with loops. *Neural Computation*, 12(1):1–41, 2000.
- A. Weller and T. Jebara. Bethe bounds and approximating the global optimum. In *Artificial Intelligence and Statistics*, 2013a.
- A. Weller and T. Jebara. On MAP inference by MWSS on perfect graphs. In *Uncertainty in Artificial Intelligence (UAI)*, 2013b.
- A. Weller and T. Jebara. Approximating the Bethe partition function. In *Uncertainty in Artificial Intelligence (UAI)*, 2014.
- M. Welling and Y. Teh. Belief optimization for binary networks: A stable alternative to loopy belief propagation. In *Uncertainty in Artificial Intelligence (UAI)*, 2001.
- T. Werner. Primal view on belief propagation. In *Uncertainty in Artificial Intelligence (UAI)*, 2010.
- J. Yedidia, W. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. In *International Joint Conference on Artificial Intelligence, Distinguished Lecture Track*, 2001.

A Bayesian Nonparametric Model for Spectral Estimation of Metastable Systems

Hao Wu

Department of Mathematics and Computer Science
Free University of Berlin
Arnimallee 6, 14195 Berlin, Germany

Abstract

The identification of eigenvalues and eigenfunctions from simulation or experimental data is a fundamental and important problem for analysis of metastable systems, because the dominant spectral components usually contain a lot of essential information of the metastable dynamics on slow timescales. It has been shown that the dynamics of a strongly metastable system can be equivalently described as a hidden Markov model (HMM) under some technical assumptions and the spectral estimation can be performed through HMM learning. However, the spectral estimation with unknown number of dominant spectra is still a challenge in the framework of traditional HMMs, and the infinite HMMs developed based on stick-breaking processes cannot satisfactorily solve this problem either. In this paper, we analyze the difficulties of spectral estimation for infinite HMMs, and present a new nonparametric model called stick-breaking half-weighted model (SB-HWM) to address this problem. The SB-HWM defines a sparse prior of eigenvalues and can be applied to Bayesian inference of dominant eigenpairs of metastable systems in a nonparametric manner. We demonstrate by simulations the advantages of applying SB-HWM to spectral estimation.

1 INTRODUCTION

In a variety of scientific areas, we are confronted with the task of analyzing and modeling a complex system which can be described as a Markov process $\{x_t\}$ with time evolution equation

$$\begin{aligned}\rho_{t+\tau}(x) &= \mathcal{P}(\tau) \rho_t(x) \\ &\triangleq \int_{\Omega} p(x_{t+\tau} = x | x_t = x') \rho_t(x') dx \quad (1)\end{aligned}$$

where x_t denotes the system state at time t , Ω is the state space, ρ_t represents the probability density function of x_t , and \mathcal{P} represents the Markov propagator. For many real-world physical and chemical systems, e.g., conformational transitions in macromolecules (Noé and Fischer, 2008), autocatalytic chemical reactions (Biancalani et al., 2012) and climate changes (Berglund and Gentz, 2002), it is common and natural to further assume that $\{x_t\}$ is a time-reversible and metastable process. The reversibility means that $p(x_t = x', x_{t+\tau} = x) = p(x_t = x, x_{t+\tau} = x')$ and generally arises from the time symmetries of classical mechanics, thermodynamics and quantum mechanics, and the metastability of a dynamical system means that the state space of the system can be decomposed into a set of macrostates called *metastable states* so that the local equilibrium within a metastable state can be reached quickly and the transitions between different metastable states can only be observed on slow timescales. A large number of recent studies in statistical physics indicate that the dominant spectral components (or called dominant eigenpairs, i.e., the largest eigenvalues and the associated eigenfunctions) of the Markov propagator is a key to understand and characterize such a process, because they can provide a lot of essential and useful information for the computation of ensemble averages and correlation functions (Noé et al., 2011), detection of spatial structures of metastable states (Deuffhard and Weber, 2005), choice of reaction coordinates (Rohrdanz et al., 2011; Perez-Hernandez et al., 2013), and construction of low-dimensional approximate models (Kube and Weber, 2007; Noé and Nüske, 2013).

However, directly solving the eigenvalue problem of the Markov propagator is generally impossible except for some extremely simple cases (e.g., Ornstein-Uhlenbeck process), and the dominant spectral components can only be estimated from simulation or experimental data through statistical inference and numerical computation. The most

popular and successful method for the spectral estimation is the Markov state model (MSM) method (Prinz et al., 2011; Djurdjevac et al., 2010), which discretizes the state space into a set of discrete bins and calculates the dominant spectral components in a finite element manner by assuming transitions between the bins are Markovian. Obviously, the main difficulty of this method is the choice of the discretization. On the one hand the Markov assumption will be severely violated if the discretization is too coarse, and on the other hand too many bins may cause the problem of “curse of dimensionality” in the estimation of transition probabilities. A more general method is the variational method (Noé and Nüske, 2013; Nüske et al., 2013), which allows one to perform the spectral estimation by using “soft bins” defined by a set of smooth basis functions instead of the “crisp bins” used in MSMs. Numerical experiments show that the variational method can achieve more accurate estimation than the MSM method with the same number of bins. However, there is no systematic algorithm for the choice of basis functions, and the basis function set can only be determined by trial and error in practice. Moreover, in some literature, the diffusion maps is used to identify dominant spectral components in a nonparametric manner by treating each sample point as a discrete bin (Rohrdanz et al., 2011; Ferguson et al., 2011), but this method is applicable only if the Markov propagator is defined by a Brownian dynamics.

In (Noé et al., 2013; Prinz et al., 2014), a novel framework call projected Markov model (PMM) is proposed for spectral analysis of metastable processes without the Markov assumption on discrete bins. Within this framework, it is shown that if a metastable Markov process contains only m nonzero eigenvalues then the corresponding coarse-grained dynamics on the space of discrete bins is equivalent to an m -state hidden Markov model (HMM), and the equivalence is independent of the choice of the discretization. Then HMM learning methods can be utilized to identify dominant eigenvalues and projected eigenfunctions efficiently and effectively even in the case that the investigated system is only experimentally observable and some important dimensions of the system state cannot be directly observed. (See more details in Subsection 2.1.) The main disadvantage of the PMM approach is that the estimation performance strongly depends on the choice of m , and a small change of the value of m may lead to a great error on the estimation of spectral components because of the orthogonality of eigenfunctions (Noé et al., 2013).

The aim of this paper is to propose an infinite HMM based method to solve the spectral estimation problem of metastable processes with unknown dominant spectra. Infinite HMMs (Teh et al., 2006; Teh and Jordan, 2010; Paisley and Carin, 2009; Fox et al., 2011) are a generalization of classical HMMs, which contains infinite hidden states and provide a powerful tool for nonparametric dynamical

modeling of sequential data. In contrast with classical HMMs, infinite HMMs encourage sparse utilization of infinite state sets through defining suitable prior models, and can be used to infer both model parameters and state numbers from observation data in a pure Bayesian manner. However, our investigation (see Section 3) shows that a sparse prior on hidden states cannot guarantee that the eigenvalue set also has a sparse structure, and the spectral estimation is an “ill-posed” problem for the existing infinite HMMs. In this paper, we construct a new infinite HMM named stick-breaking half-weighted model (SB-HWM), which has a sparse prior distribution on eigenvalues and tends to approximate the underlying dynamics of an unknown system with a small number of dominant spectral components. Moreover, we develop a sampling inference algorithm for applying SB-HWMs to Bayesian nonparametric inference of spectral components.

The rest of the paper is organized as follows. In Section 2, we present the relevant mathematical background on PMMs and infinite HMMs, then Section 3 outlines the Bayesian nonparametric framework for solving the spectral estimation problem and explains the reason why the existing infinite HMMs cannot be directly applied. In Section 4 we introduce the SB-HWM and its sampling inference algorithm. Section 5 demonstrates through simulations the effectiveness of the proposed model and algorithm.

2 BACKGROUND

2.1 COARSE-GRAINED DYNAMICS AND PROJECTED MARKOV MODELS

Let $\{x_t\}$ be a Markov process with propagator \mathcal{P} and state space Ω as in (1) and $\{y_t\}$ is the corresponding observation process obtained from the spatial coarse-graining

$$\Pr(y_t = k | x_t = x) = \chi_k(x), \quad k \in \mathcal{O} \quad (2)$$

where $\mathcal{O} = \{1, \dots, K\}$ denotes the discrete observation space and $\chi_k(x)$ denotes the observation probability function for the observed value k . Often, the coarse-graining is employed by the Galerkin discretization and $\{\chi_k(x)\}$ is a set of indicator functions with each k representing a finite element space $\{x | x \in \Omega, \chi_k(x) = 1\}$. But in some practical cases, e.g. where $\{y_t\}$ obtained from noisy measurements, each $\chi_k(x)$ is a continuous probability density function and characterizes a soft finite element space.

It is obvious that (1) and (2) is in fact an HMM, but it is infeasible to reconstruct \mathcal{P} from $\{y_t\}$ by direct statistical inference because of the continuity of Ω and the complexity of the dynamics of $\{x_t\}$ in general cases. In order to overcome this difficulty, the PMM (Noé et al., 2013) provides a low-dimensional approximation of the coarse-grained dynamics based on the following metastability assumption:

Assumption 1. $\{x_t\}$ is ergodic and reversible w.r.t. the

unique stationary distribution $\mu(x)$, and there is a τ' such that $\mathcal{P}(\tau')$ has only m eigenvalues which are not close to 0.

Note that this assumption holds for most practical metastable systems and m is usually a small number depends on the number of metastable states in Ω .¹ Under this assumption, we can conclude that $\mathcal{P}(\tau)$ is a compact and self-adjoint operator w.r.t. the inner product inner product $\langle \cdot, \cdot \rangle_{\mu^{-1}}$ defined by

$$\langle u_1, u_2 \rangle_{\mu^{-1}} = \int \frac{u_1(x) u_2(x)}{\mu(x)} dx \quad (3)$$

and the dynamics of $\{x_t\}$ can be decomposed as

$$\rho_{t+\tau} = \sum_{i=1}^m \lambda_i(\tau) \langle \rho_t, \phi_i \rangle_{\mu^{-1}} \phi_i + \mathcal{P}_{\text{fast}}(\tau) \rho_t \quad (4)$$

with

$$\lambda_i(\tau) = \exp(-\kappa_i \tau) \quad (5)$$

Here $\lambda_i(\tau)$ denotes the i -th largest magnitude eigenvalue of $\mathcal{P}(\tau)$ with eigenfunction ϕ_i and decay rate $\kappa_i \geq 0$ ($\kappa_1 = 0 < \kappa_2$ and $\phi_1 = \mu$ due to the ergodicity). The operator $\mathcal{P}_{\text{fast}}(\tau)$ consists of spectral components of $\mathcal{P}(\tau)$ which decay to zero quickly and $\|\mathcal{P}_{\text{fast}}(\tau)\| \approx 0$ for $\tau \geq \tau'$. Omitting the second term on the r.h.s. of (4), the correlation matrix $\mathbf{C}(n\tau) = [c_{ij}(n\tau)] = [\Pr(y_t = i, y_{t+n\tau} = j)]$ of $\{y_t\}$ can be decomposed as

$$\mathbf{C}(n\tau) = \mathbf{Q} \mathbf{\Lambda}(\tau)^n \mathbf{Q}^\top \quad (6)$$

where $\mathbf{\Lambda}(\tau) = \text{diag}(\lambda_1(\tau), \dots, \lambda_m(\tau))$ contains the dominant eigenvalues of $\mathcal{P}(\tau)$, and the i -th column of $\mathbf{Q} \in \mathbb{R}^{K \times m}$ is the i -th projected eigenfunction

$$\mathbf{q}_i = \left(\int \chi_1(x) \phi_i(x) dx, \dots, \int \chi_K(x) \phi_i(x) dx \right)^\top \quad (7)$$

Therefore, we can characterize the coarse-grained dynamics of $\{y_t\}$ by low-dimensional PMM variables $\{\mathbf{Q}, \mathbf{\Lambda}(\tau)\}$ on a large timescale $\tau \geq \tau'$.

It is important to point out that we can also get a similar approximation of $\mathbf{C}(n\tau)$ by using a m -state HMM. Assume that $\{y_t\}$ are observations of an HMM with hidden states $\{s_t\}$, state set $\{1, \dots, m\}$, transition matrix $\mathbf{A} = [a_{ij}] = [\Pr(s_{t+\tau} = j | s_t = i)]$ and observation matrix $\mathbf{B} = [b_{ij}] = [\Pr(y_t = j | s_t = i)]$, then $\mathbf{C}(n\tau)$ can be

¹Generally speaking, a stochastic system with m metastable states only has m eigenvalues which are significantly larger than zero on a large timescale. It is worth pointing out that this assumption of sparse spectrum is a very important basis in the research of metastability (see, e.g., (Deuffhard and Weber, 2005; Djurdjevac et al., 2010; Noé and Nüske, 2013; Prinz et al., 2014))), and a large number of studies have shown the validity of this assumption for common physical processes which exhibits metastability.

expressed as

$$\begin{aligned} \mathbf{C}(n\tau) &= \mathbf{B}^\top \text{diag}(\boldsymbol{\pi}) \mathbf{A}^n \mathbf{B} \\ &= (\mathbf{B}^\top \mathbf{L}) \tilde{\mathbf{\Lambda}}^n (\mathbf{B}^\top \mathbf{L})^\top \end{aligned} \quad (8)$$

under the condition² that \mathbf{A} is a reversible transition matrix w.r.t. the stationary distribution $\boldsymbol{\pi}$, where $\tilde{\mathbf{\Lambda}}$ is a diagonal matrix containing eigenvalues of \mathbf{A} , \mathbf{L} consists of left eigenvectors of \mathbf{A} with $\mathbf{L}^\top \mathbf{A} = \tilde{\mathbf{\Lambda}} \mathbf{L}^\top$ and $\mathbf{L}^\top \text{diag}(\boldsymbol{\pi})^{-1} \mathbf{L} = \mathbf{I}$, and \mathbf{I} denotes the identity matrix. Based on the similarity between (6) and (8), the PMM theory provides the following conclusion: *Under the metastability assumption (Assumption 1) with $\|\mathcal{P}_{\text{fast}}(\tau)\| = 0$ and some technical assumptions, the dynamics of $\{y_t\}$ is equivalent to a m -state HMM with a reversible transition matrix.* Thus, if a suitable m is given, we can utilize HMM learning algorithms to efficiently estimate the dominant eigenvalues and projected eigenfunctions of $\mathcal{P}(\tau)$ from $\{y_t\}$ with $\mathbf{Q} = \mathbf{B}^\top \mathbf{L}$ and $\mathbf{\Lambda}(\tau) = \tilde{\mathbf{\Lambda}}$. However, the choice of m is still an unsatisfactorily solved problem for the PMM method, and the numerical experiments in (Noé et al., 2013) show that the estimation results of the PMM method is very sensitive to the value of m .

2.2 STICK-BREAKING PROCESSES AND INFINITE HIDDEN MARKOV MODELS

Roughly speaking, a stick-breaking process (SBP) (Ishwaran and James, 2001) is a prior for discrete distributions, and the realization of an SBP with parameters α', α and base distribution G_0 can be expressed by the following probability density function:

$$G = \sum_{i=1}^{\infty} w_i \delta_{\theta_i} \quad (9)$$

where δ_{θ} denotes the Dirac point measure concentrated on θ , θ_i is the i -th component of the discrete distribution with $\theta_i \stackrel{\text{iid}}{\sim} G_0$, and w_i denotes the corresponding weight which are drawn by $w_i = V_i \prod_{j=1}^{i-1} (1 - V_j)$ and $V_i \stackrel{\text{iid}}{\sim} \text{Beta}(\alpha', \alpha)$. Obviously, the SBP model is a generalization of the finite-dimensional Dirichlet distribution (Gelman et al., 2003), and allows one to easily construct discrete distributions with infinite components. For convenience of computation and notation, in this paper we only consider a special class³ of SBPs with $\alpha' = 1$, and denote by DP(α, G_0) and GEM(α) the prior distributions of G and $\{w_i\}$ defined in (9).

The SBP model provides a powerful and flexible tool for nonparametric estimation of multi-modal mixture models, and can be applied to building HMMs with infinite

²This condition means $\text{diag}(\boldsymbol{\pi}) \mathbf{A}$ is a symmetric matrix, which is a sufficient condition for reversibility of $\{y_t\}$.

³An SBP with $\alpha' = 1$ is equivalent to a Dirichlet process (Ferguson, 1973).

states for sequential statistical modeling. The most commonly used infinite HMM is the HDP-HMM (Teh et al., 2006), which constructs prior distributions of the infinite-dimensional transition matrix $\mathbf{A} = [a_{ij}]$ and observation matrix $\mathbf{B} = [b_{ij}]$ by organizing multiple SBPs in a hierarchical structure as

$$\begin{aligned} G_0 &= \sum_{k=1}^{\infty} \beta_k \delta_{\mathbf{b}_k} \sim \text{DP}(\gamma, H) \\ G_i &= \sum_{j=1}^{\infty} a_{ij} \delta_{\mathbf{b}_j} \stackrel{\text{iid}}{\sim} \text{DP}(\alpha, G_0) \end{aligned} \quad (10)$$

where \mathbf{b}_i denotes the i -th row of \mathbf{B} and represents the observation probability distribution of the i -th state, H represents the prior distribution of each \mathbf{b}_i and is usually a Dirichlet distribution, and α, γ are hyperparameters. In (Fox et al., 2011), a modified HDP-HMM called “sticky HDP-HMM” is proposed to encourage large self-transition probabilities and avoids “unphysically” fast switching between different states, which can be expressed as

$$\begin{aligned} G_0 &= \sum_{k=1}^{\infty} \beta_k \delta_{\mathbf{b}_k} \sim \text{DP}(\gamma, H) \\ G_i &= \sum_{j=1}^{\infty} a_{ij} \delta_{\mathbf{b}_j} \stackrel{\text{ind}}{\sim} \text{DP}(\alpha + \kappa, G_0^{(i)}) \end{aligned} \quad (11)$$

with

$$G_0^{(i)} = \frac{\alpha G_0 + \kappa \delta_{\mathbf{b}_i}}{\alpha + \kappa} \quad (12)$$

where $\kappa > 0$ is the sticky factor and $\lim_{\kappa \rightarrow \infty} a_{ii} = 1$. Furthermore, it is worthwhile to point out that for most of the SBP based infinite HMMs, including HDP-HMM, sticky HDP-HMM and stick-breaking HMM proposed in (Paisley and Carin, 2009), the infinite-dimensional prior distributions can be approximated by high- but finite-dimensional ones for convenience of implementing sampling inference.

3 BAYESIAN NONPARAMETRIC FRAMEWORK FOR SPECTRAL ESTIMATION

The main purpose of this paper is to develop a Bayesian nonparametric framework for spectral estimation of metastable Markov processes with unknown number m of dominant eigenpairs. In the rest of paper, unless otherwise stated, the lagtime τ is set to be fixed, and $\{x_t\}$ and $\{y_t\}$ are separately defined as $\{x_{n\tau}\}_{n=0}^N$ and $\{y_{n\tau}\}_{n=0}^N$.

Suppose that $\{x_t\}$ is a metastable process with the available observation process $\{y_t\}$ as described in Subsection 2.1 and $\{x_t\}$ satisfies Assumption 1. Based on the discussion in Section 2, the Bayesian estimation of the i -th largest eigenvalue λ_i and the corresponding projected eigenfunction \mathbf{q}_i of the Markov propagator $\mathcal{P}(\tau)$ of $\{x_t\}$ can be achieved by the following steps with m not given a priori: First, the dynamics of $\{y_t\}$ is described by an infinite HMM consisting of an infinite-dimensional transition matrix \mathbf{A} and observation matrix \mathbf{B} with prior $p(\mathbf{A}, \mathbf{B})$. Second, a large number of samples $\{(\mathbf{A}^{(k)}, \mathbf{B}^{(k)})\}$ of (\mathbf{A}, \mathbf{B}) are

drawn from the posterior distribution

$$\begin{aligned} p(\mathbf{A}, \mathbf{B} | \{y_t\}) \\ \propto p(\mathbf{A}, \mathbf{B}) \sum_{\{s_t\}} p(\{s_t\} | \mathbf{A}) p(\{y_t\} | \{s_t\}, \mathbf{B}) \end{aligned} \quad (13)$$

where s_t denotes the discrete hidden state of the infinite HMM at time t . Finally, the i -th eigenvalue $\lambda_i^{(k)}$ and left eigenvector $\mathbf{l}_i^{(k)\top}$ of $\mathbf{A}^{(k)}$ are calculated for each k such that the posterior distribution of $(\lambda_i, \mathbf{q}_i)$ can be approximated by the ensemble $\{(\lambda_i^{(k)}, \mathbf{q}_i^{(k)})\}$ with $\mathbf{q}_i^{(k)} = \mathbf{B}^{(k)\top} \mathbf{l}_i^{(k)}$.

It is natural for us to utilize one of infinite HMMs such as the HDP-HMM and sticky HDP-HMM mentioned in Subsection 2.2 to design the prior distribution of (\mathbf{A}, \mathbf{B}) within the above framework. However, the following simple example shows that the existing SBP based infinite HMMs are *not* applicable to the spectral estimation problem.

Example 2. Let $\{s_t\} = \{s_{n\tau}\}_{n=0}^{1000}$ be a realization of a reversible 3-state Markov chain with transition matrix

$$\mathbf{A}_0 = \begin{bmatrix} 0.8462 & 0.0769 & 0.0769 \\ 0.1250 & 0.7500 & 0.1250 \\ 0.1818 & 0.1818 & 0.6364 \end{bmatrix} \quad (14)$$

It is clear that $\{s_t\}$ is a Markov chain with large self-transition probabilities and has only three nonzero eigenvalues. We use the prior models of infinite-dimensional transition matrices defined in the HDP-HMM and sticky HDP-HMM to approximate the first 5 eigenvalues of $\{s_t\}$ based on the posterior distribution

$$\begin{aligned} p(\mathbf{A} | \{s_t\}) &\propto p(\mathbf{A}) p(\{s_t\} | \mathbf{A}) \\ &\propto p(\mathbf{A}) \prod_{n=1}^{1000} a_{s_{(n-1)\tau}, s_{n\tau}} \end{aligned} \quad (15)$$

Fig. 1a illustrates the estimation results obtained by the Markov chain Monte Carlo (MCMC) sampling. It can be observed that both the HDP-HMM and the sticky HDP-HMM give poor estimates of eigenvalues and fail to detect the spectral gap between λ_3 and λ_4 .

In a strict sense, the estimation problem in Example 2 is not an “HMM problem” since the hidden state sequence $\{s_t\}$ is exactly known, rather, it is an effective toy example for illustrating the difficulty of nonparametric spectral estimation for the existing infinite HMMs. Roughly speaking, each infinite HMM provides a “sparse” prior distribution of the stationary distribution $\pi = [\pi_i]$ of the transition matrix \mathbf{A} , which means for most samples of π we can find a small set \mathcal{S} of hidden states such that $\sum_{i \notin \mathcal{S}} \pi_i \approx 0$. Thus, there are only a small number of distinct hidden states that can be detected by the Bayesian inference in general although the prior model contains infinite states. However, the sparsity of π cannot guarantee the sparsity of the eigenvalue set, because the transition dynamics between hidden

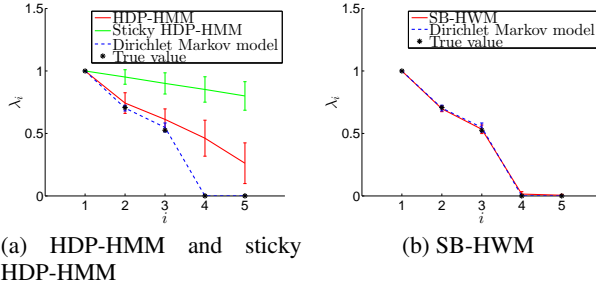


Figure 1: Estimation results of the first 5 eigenvalues of $\{s_t\}$ based on different prior models of \mathbf{A} , where error bars represent one standard deviation confidence intervals, and dashed lines represent the estimation results obtained by assuming that \mathbf{A} is a transition matrix with size 3×3 and the prior of each row of \mathbf{A} is $\text{Dir}(1/3, 1/3, 1/3)$. Truncated-model-based samplers (see Subsection 4.3 and (Fox et al., 2008)) are applied to sampling \mathbf{A} of infinite HMMs, which approximate \mathbf{A} by a finite-dimensional matrix with size 20×20 , and the posterior means and standard deviations are calculated from 5000 MCMC samples with 5000 burn-in samples.

states with small stationary probabilities may also contain large eigenvalues. This is also the reason why the infinite HMMs overestimate the eigenvalues in Example 2. (In fact, for any stationary distribution π and $i > 0$ we can construct a sequence of matrices $\{\mathbf{A}^{(k)}\}$ which are reversible w.r.t. π and satisfy $\lim_{k \rightarrow \infty} \lambda_i^{(k)} \rightarrow 1$. See the supplementary material for details.) Note that in contrast with the HDP-HMM, the sticky HDP-HMM encourages longer residence time for each state and tends to generate more “pseudo-dominant eigenvalues”, so it performs worse than the HDP-HMM in this example. Furthermore, it is difficult for both HDP-HMM and sticky HDP-HMM to incorporate the reversibility constraint.

In order to overcome disadvantages of existing infinite HMMs in the application of spectral estimation, we present in next section a novel infinite HMM, which approximates the “half-weighted matrix” instead of the transition matrix in a nonparametric way and can provide a sparse prior for eigenvalues.

4 STICK-BREAKING HALF-WEIGHTED MODELS

4.1 HALF-WEIGHTED MATRICES

Before developing our infinite HMM for spectral estimation, we first introduce the definition and some important properties of half-weighted matrices for the purpose of self-containedness. For a Markov chain with transition matrix $\mathbf{A} = [a_{ij}]$ and stationary distribution $\pi = [\pi_i]$, the half-

weighted matrix $\mathbf{H} = [h_{ij}]$ is defined by⁴

$$\mathbf{H} = \text{diag}(\pi)^{\frac{1}{2}} \mathbf{A} \text{diag}(\pi)^{-\frac{1}{2}} \quad (16)$$

(Note dimensions of \mathbf{A} , π and \mathbf{H} may be infinite here.)

The following two theorems summarize important properties of the half-weighted matrix and provide a criterion for checking if a matrix is a valid half-weighted matrix. (The proofs are in the supplementary material.)

Theorem 3. *If \mathbf{A} is a reversible and positive transition matrix and all eigenvalues $\{\lambda_i\}$ of \mathbf{A} are square summable, then (1) \mathbf{H} is a positive and symmetric matrix. (2) $\|\mathbf{H}\|_F < \infty$. (3) \mathbf{H} and \mathbf{A} have the same eigenvalues, and the i -th eigenvector ψ_i of \mathbf{H} and the i -th left eigenvector \mathbf{l}_i of \mathbf{A} satisfy $\psi_i = \text{diag}(\pi)^{-\frac{1}{2}} \mathbf{l}_i$. (4) $\sum_{i=m+1}^{\infty} \lambda_i^2 \leq \sum_{i>m} \sum_{j>m} h_{ij}^2$ for all $m > 1$.*

Theorem 4. *If \mathbf{H} is a positive and symmetric matrix with $\|\mathbf{H}\|_F < \infty$, and the spectral radius of \mathbf{H} is 1, then \mathbf{H} is a half-weighted matrix of a Markov chain.*

According to (16), the likelihood of a half-weighted matrix \mathbf{H} of a given state sequence $\{s_t\} = \{s_{n\tau}\}_{n=0}^N$ is

$$p(\{s_t\}|\mathbf{H}) = p(s_0) \sqrt{\frac{\pi_{s_{N\tau}}}{\pi_{s_0}}} \prod_{n=1}^N h_{s_{(n-1)\tau}, s_{n\tau}} \quad (17)$$

From the above, it can be seen that the half-weighted matrix \mathbf{H} can be used to describe the dynamics of state transitions instead of \mathbf{A} , and \mathbf{H} is more numerically stable for eigenvalue decomposition than \mathbf{A} due to the symmetry of \mathbf{H} . Moreover, it is interesting to observe that (17) is in fact a Boltzmann chain model (Saul and Jordan, 1995) with the transition energy from state i to state j being $-\ln h_{ij}$.

4.2 MODEL DEFINITION

From the fourth property of half-weighted matrices stated in Theorem 3, it can be seen that if a Markov chain has a half-weighted matrix with all elements except the ones in a small number of rows and columns are close to zero, then there are only a few eigenvalues of the Markov chain that can be significantly larger than zero. This suggests a natural way of constructing a prior distribution over infinite HMMs which encourages the sparsity of eigenvalue sets and satisfies the reversibility.

Based on the above discussion, we now propose the following infinite HMM called stick-breaking half-weighted model (SB-HWM) for spectral estimation:

$$\begin{aligned} \mathbf{H} &= \frac{1}{r} \tilde{\mathbf{H}} \\ \mathbf{B} &= (\mathbf{b}_1^\top, \mathbf{b}_2^\top, \dots)^\top \end{aligned} \quad (18)$$

⁴The definition of half-weighted matrix is in fact a discrete version of the “half-weighted correlation density” proposed in (Noé and Nüske, 2013) for analysis of Markov processes.

with

$$\begin{aligned}
\mathbf{w} = [w_i] &\sim \text{GEM}(\alpha_w) \\
\gamma_{ij} &\stackrel{\text{iid}}{\sim} \text{Gamma}(\alpha_\gamma, \beta_\gamma), \quad \text{for } i \geq j \\
\gamma_{ji} &= \gamma_{ij}, \quad \text{for } i < j \\
w_i^d &\stackrel{\text{iid}}{\sim} \text{Gamma}(\alpha_d, \beta_d) \\
\bar{\mathbf{H}} = [\bar{h}_{ij}] &= [\gamma_{ij} (w_i w_j + w_i w_i^d \cdot 1_{i=j})] \quad (19)
\end{aligned}$$

and

$$\mathbf{b}_i \stackrel{\text{iid}}{\sim} \text{Dir}(\alpha_b, \dots, \alpha_b) \quad (20)$$

where r denotes the spectral radius of the “unnormalized half-weighted matrix” $\bar{\mathbf{H}}$, $(\alpha_w, \alpha_\gamma, \beta_\gamma, \alpha_d, \beta_d, \alpha_b)$ are hyperparameters, and it is easy to verify by Theorem 4 that the realization of $\bar{\mathbf{H}}$ is a valid half-weighted matrix with probability 1. Note that $\bar{\mathbf{H}}$ can be expressed in a more compact form as

$$\bar{\mathbf{H}} = \mathbf{\Gamma} \circ (\mathbf{w}\mathbf{w}^\top + \text{diag}(\mathbf{w} \circ \mathbf{w}^d)) \quad (21)$$

where $\mathbf{\Gamma} = [\gamma_{ij}]$, $\mathbf{w}^d = [w_i^d]$ and \circ denotes the element-wise product. It can be seen that \mathbf{w} employs a “template vector” to encourage rows and columns of the half-weighted matrix to have the similar sparse structures⁵. Furthermore, it is known that for a metastable Markov process, the hidden states of the equivalent HMM mentioned in Subsection 2.1 often have long residence times since they arise from metastable states of the original process (Noé et al., 2013). So we use \mathbf{w}^d to enhance probabilities of self-transitions of hidden states, which plays the similar role as the sticky factor κ in (11). The following theorem gives a theoretical description of the sparsity of π and eigenvalue set $\{\lambda_i\}$ in the SB-HWM (see the supplementary material for the proof):

Theorem 5. *For an SB-HWM (\mathbf{H}, \mathbf{B}) generated by the prior defined by (18)-(20), the i -th largest magnitude eigenvalue λ_i and the stationary probability π_i of the i -th hidden satisfy $\mathbb{E}[|\lambda_i|] = O\left(\left(\frac{\alpha_w}{1+\alpha_w}\right)^{\frac{1}{3}}\right)$ and $\mathbb{E}[\pi_i] = O\left(\left(\frac{\alpha_w}{1+\alpha_w}\right)^{\frac{1}{3}}\right)$ as $i \rightarrow \infty$.*

As a comparison, we also apply the SB-HWM to the data in Example 2 and the estimation results are shown in Fig. 1b. (See Subsection 4.3 for the sampling algorithm.) It can be seen that the SB-HWM achieves the similar estimation performance as the HMM with state number given, and the correct number of dominant spectral components can be easily obtained from samples of the SB-HWM.

⁵For example, if w_i is about zero, the elements in the i -th row and column of $\bar{\mathbf{H}}$ will also be close to zero with high probabilities.

4.3 SAMPLING INFERENCE

For convenience of computation, we first construct a truncated model to approximate the SB-HWM by replacing the prior distribution of \mathbf{w} in (19) with the following truncated SBP prior (Ishwaran and James, 2002):

$$w_i = V_i \prod_{j=1}^{i-1} (1 - V_j) \quad (22)$$

with

$$\begin{cases} V_i \stackrel{\text{iid}}{\sim} \text{Beta}(1, \alpha), & i < L \\ V_i = 1, & i = L \end{cases} \quad (23)$$

The truncated model is obviously a finite HMM with L states since $w_i = 0$ for $i > L$, and according to Theorem 5, the influence of the truncation on the dynamics of SB-HWM is slight if L is sufficiently large⁶. Then the Markov chain Monte Carlo approach can be utilized to draw samples of (\mathbf{H}, \mathbf{B}) from the posterior distribution $p(\mathbf{H}, \mathbf{B} | \{y_t\})$ based on the truncated prior. Considering that the presented prior distribution of \mathbf{H} is not a conjugate distribution for the state sequence, here we combine the Metropolis-within-Gibbs algorithm with the block sampling algorithm of classical HMMs to generate samples (see the supplementary material for details) based on the assumption that $\{s_t\}$ is a stationary process, i.e., $p(s_0) = \pi_{s_0}$.

5 APPLICATIONS

In this section, we demonstrate the performance of the SB-HWM based Bayesian spectral estimation method on three examples of stochastic systems including an HMM, a diffusion process governed by a Brownian dynamics and the molecular dynamics of alanine dipeptide. The detailed settings of simulations and estimation algorithms are provided in the supplementary material.

5.1 HMM DATA

Here we apply the SB-HWM to the simulation data generated by a 3-state HMM with lagtime $\tau = 1$ and 8, and compare its performance with HDP-HMM and sticky HDP-HMM. Estimation results are summarized in Fig. 2. Obviously, both HDP-HMM and sticky HDP-HMM severely overestimate the eigenvalues and result in large errors in estimation of projected eigenfunctions, because their samples contain a lot of “pseudo-dominant spectral components” as mentioned in Section 3. (All the three models achieve small

⁶According to our experience, the empirical performance of truncated SB-HWMs is not sensitive to the choice of L if it is larger than twice or three times of the number of the dominant eigenvalues. We simply set L to be 20 in experiments of this paper, and the theoretical analysis of the truncation error will be published elsewhere.

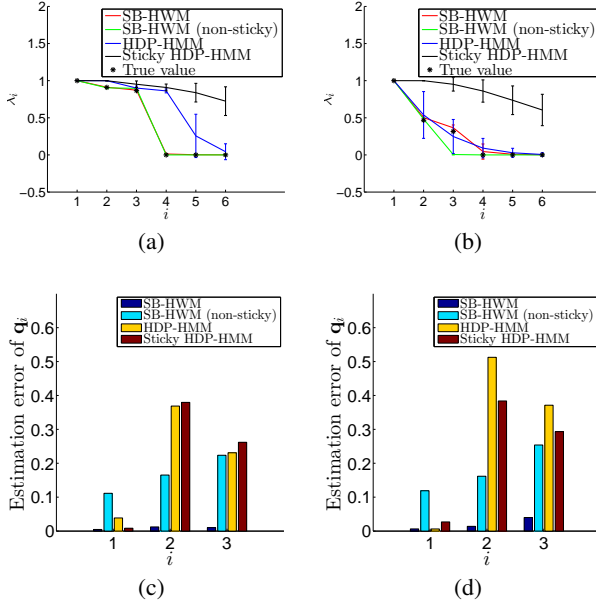


Figure 2: Infinite HMMs applied to data from an HMM. (a,b) Estimates of the first 6 eigenvalues with $\tau = 1$ and 8, where error bars represent one standard deviation confidence intervals. (c,d) Estimation errors of the first 3 projected eigenfunctions with $\tau = 1$ and 8, where the error between the estimate \hat{q}_i and the true value q_i is defined by $\|\hat{q}_i - q_i\|$.

estimation errors on q_1 as it can easily be estimated as the stationary distribution of $\{y_t\}$. Of the above three models, only the SB-HWM provides accurate estimates of eigenvalues and dominant eigenfunctions, which allows us to correctly detect the spectral gap and total number of dominant spectral components. Furthermore, we apply a specific SB-HWM with $w_i^d \equiv 0$ for all i to the HMM data in order to verify the usefulness of the sticky term in (19), and the estimates obtained by the non-sticky SB-HWM are also shown in Fig. 2 (see green lines and bars). It can be observed that the estimates of projected eigenfunctions obtained by the non-sticky SB-HWM are much worse than that obtained by the proposed SB-HWM and the non-sticky SB-HWM fails to identify the third dominant spectral component when applied to the HMM data with $\tau = 8$. The main reason for the poor performance of the non-sticky SB-HWM is that it tends to underestimate residence times of hidden states which are key parameters affecting the spectral properties especially for HMMs of metastable systems. (Note that $a_{ii} = O(w_i^2)$ as $i \rightarrow \infty$ in the non-sticky SB-HWM, whereas $a_{ii} = O(w_i)$ in the SB-HWM.)

5.2 BROWNIAN DYNAMICS DATA

In this subsection, we consider a two-dimensional system of Brownian dynamics on the domain $\Omega = [-2, 2] \times$

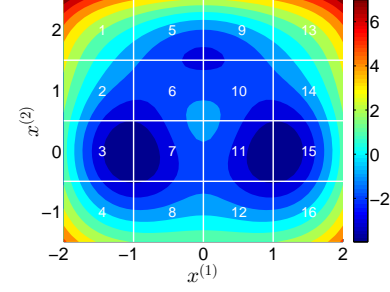


Figure 3: Illustration of the potential function and observation model of a Brownian dynamics system, where each grid represents a bin of the observation model.

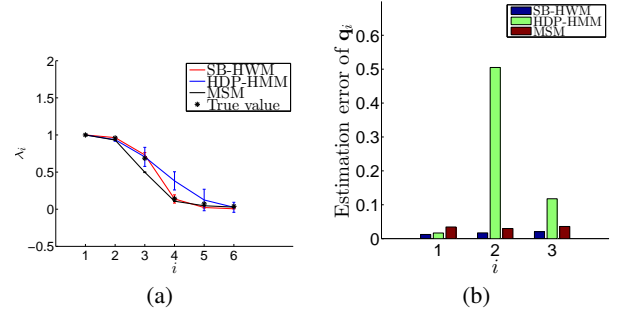


Figure 4: Infinite HMMs and an MSM applied to data from a Brownian dynamics simulation. (a) Estimates of the first 6 eigenvalues, where error bars represent one standard deviation confidence intervals. (b) Estimation errors of the first 3 projected eigenfunctions.

$[-1.5, 2.5]$ with a three-well potential and a Galerkin discretization observation model which are depicted in Fig. 3. The three potential wells implies that the system contains the same number of metastable states and dominant spectral components. Fig. 4 plots spectral estimation results obtained by the SB-HWM, HDP-HMM and MSM, where the MSM estimates spectral components by simply assuming that each bin in Fig. 3 is a discrete state in a Markov chain. It is obvious that the discrete bins cannot accurately capture boundaries between the metastable states in this example, and the poor coarse-graining causes large estimation errors of eigenvalues. (The detailed theoretical analysis on the relationship between the spectral estimation error and the choice of the discretization is reported in (Sarich et al., 2010).) From Fig. 4, we can also see that the HDP-HMM performs even much worse than the simple MSM, which again demonstrates the difficulty of spectral estimation for the existing infinite HMMs. The SB-HWM significantly outperforms the other two models on the spectral estimation in this example.

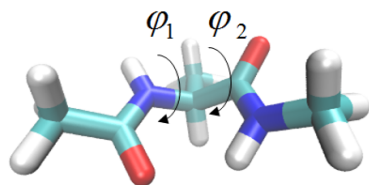


Figure 5: Illustration of the structure of alanine dipeptide

5.3 MOLECULAR DYNAMICS DATA

Alanine dipeptide (sequence acetyl-alanine-methylamide) is a small molecule which consists of two alanine amino acid units. The structural and dynamical properties of this molecule have been thoroughly studied, and it is well known that the configuration space of the alanine dipeptide can be conveniently described by two backbone dihedral angles (see Fig. 5) and contains three metastable states (see Fig. 6). We utilize the SB-HWM, HDP-HMM, 5-state MSM and 23-state MSM to perform the spectral estimation based on a molecular dynamics simulation with length 0.05 millisecond, where the discretization of all models are designed by using the kmeans algorithm and the first three models share the same discretization shown in Fig. 6. Moreover, for convenience of comparison, we construct a very finely discretized MSM with 129 states to estimate spectral components from a molecular dynamics simulation with length 1 millisecond, and use the corresponding estimates as “true values” in this example. It can be observed from Fig. 7 that the HDP-HMM cannot provide any valuable information on spectral components in this example except the first component, and the SB-HWM with observation space $\{1, \dots, 5\}$ obviously outperforms the MSMs with 5 states and 23 states.

Fig. 8 shows the estimated eigenfunctions calculated according to the estimated projected eigenfunctions provided by the SB-HWM and 129-state MSM respectively. (The calculation details are give in the supplementary material.) By comparing them, it is interesting to note that the SB-HWM is able to well reconstruct dominant eigenfunctions in a low-dimensional function space, which also demonstrates the effectiveness of the proposed spectral estimation method.

6 CONCLUSION

We introduce in this paper a novel infinite HMM, “stick-breaking half-weighted model” (SB-HWM) for identification of dominant spectral components of metastable systems. The main idea is to construct a SBP based infinite-dimensional half-weighted matrix to describe transition dynamics of hidden states. In contrast with the other infinite HMMs, the SB-HWM provides a sparse prior on eigen-

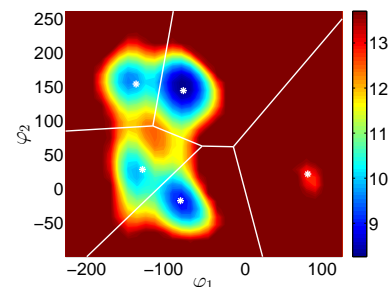


Figure 6: Free energy landscape in the state space of alanine dipeptide, where each grid represents a bin of the observation model.

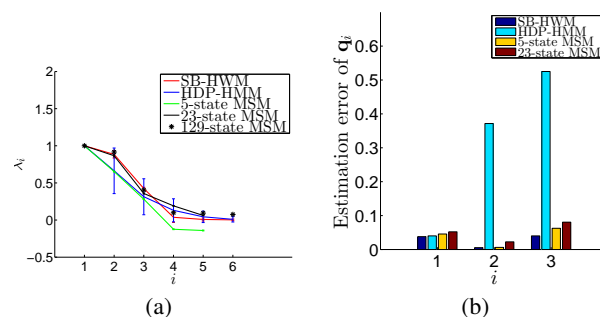


Figure 7: Infinite HMMs and MSMs applied to molecular dynamics data. (a) Estimates of the first 6 eigenvalues, where error bars represent one standard deviation confidence intervals. (Note the 5-state MSM has at most 5 nonzero eigenvalues.) (b) Estimation errors of the first 3 projected eigenfunctions.

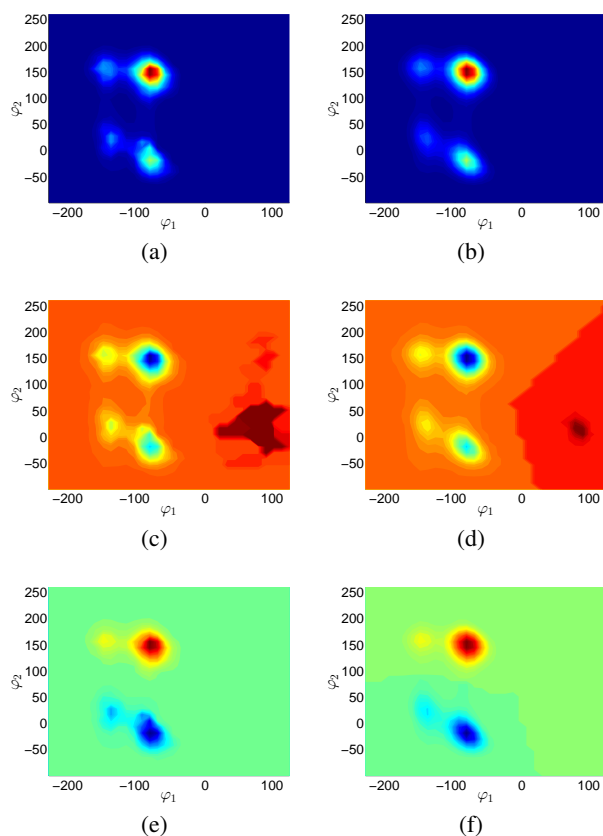


Figure 8: Estimates of the first three eigenfunctions l_1, l_2, l_3 . (a,c,e) Estimates obtained by the SB-HWM. (b,d,f) Estimates obtained by the 129-state MSM.

values so that both the values and the numbers of dominant spectral components can be estimated by the Bayesian non-parametric inference. Furthermore, a truncated approximation based sampling inference algorithm for SB-HWMs is developed. Interesting directions of future research include developing a more efficient sampling algorithm and extending the algorithm to non-reversible systems.

ACKNOWLEDGMENT

This work is supported by the Deutsche Forschungsgemeinschaft (DFG) under grant No. WU 744/1-1.

References

- Berglund, N. and B. Gentz, 2002: Metastability in simple climate models: Pathwise analysis of slowly driven langevin equations. *Stochastics and Dynamics*, **2** (03), 327–356.
- Biancalani, T., T. Rogers, and A. McKane, 2012: Noise-induced metastability in biochemical networks. *Physical Review E*, **86** (1), 010 106.
- Deuffhard, P. and M. Weber, 2005: Robust perron cluster analysis in conformation dynamics. *Linear algebra and its applications*, **398**, 161–184.
- Djurdjevac, N., M. Sarich, and C. Schütte, 2010: On markov state models for metastable processes. *Proceedings of the International Congress of Mathematicians*, Hyderabad, 3105–3131.
- Ferguson, A. L., A. Z. Panagiotopoulos, P. G. Debenedetti, and I. G. Kevrekidis, 2011: Integrating diffusion maps with umbrella sampling: Application to alanine dipeptide. *Journal of Chemical Physics*, **134** (13), 135 103.
- Ferguson, T. S., 1973: A Bayesian analysis of some non-parametric problems. *The annals of statistics*, 209–230.
- Fox, E. B., E. B. Sudderth, M. I. Jordan, and A. S. Willsky, 2008: An HDP-HMM for systems with state persistence. *Proceedings of the 25th international conference on Machine learning*, ACM, 312–319.
- Fox, E. B., E. B. Sudderth, M. I. Jordan, and A. S. Willsky, 2011: A sticky HDP-HMM with application to speaker diarization. *Annals of Applied Statistics*, **5** (2A), 1020–1056.
- Gelman, A., J. Carlin, H. Stern, and D. Rubin, 2003: *Bayesian data analysis*. 2d ed., CRC, Boca Raton.
- Ishwaran, H. and L. James, 2002: Approximate dirichlet process computing in finite normal mixtures: Smoothing and prior information. *Journal of Computational and Graphical Statistics*, **11** (3), 1–26.

- Ishwaran, H. and L. F. James, 2001: Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, **96** (453), 161–173.
- Kube, S. and M. Weber, 2007: A coarse graining method for the identification of transition rates between molecular conformations. *Journal of Chemical Physics*, **126** (2), 024 103.
- Noé, F., S. Doose, I. Daidone, M. Löllmann, M. Sauer, J. D. Chodera, and J. C. Smith, 2011: Dynamical fingerprints for probing individual relaxation processes in biomolecular dynamics with simulations and kinetic experiments. *Proceedings of the National Academy of Sciences*, **108** (12), 4822–4827.
- Noé, F. and S. Fischer, 2008: Transition networks for modeling the kinetics of conformational change in macromolecules. *Current opinion in structural biology*, **18** (2), 154–162.
- Noé, F. and F. Nüske, 2013: A variational approach to modeling slow processes in stochastic dynamical systems. *SIAM Journal on Multiscale Modeling and Simulation*, **11** (2), 635–655.
- Noé, F., H. Wu, J.-H. Prinz, and N. Plattner, 2013: Projected and hidden Markov models for calculating kinetics and metastable states of complex molecules. *Journal of Chemical Physics*, **139** (18), 184 114.
- Nüske, F., B. G. Keller, A. S. M. Guillermo Pérez-Hernández, and F. Noé, 2013: Variational approach to molecular kinetics. *Journal of Chemical Theory and Computation*.
- Paisley, J. and L. Carin, 2009: Hidden Markov models with stick-breaking priors. *IEEE Transactions on Signal Processing*, **57** (10), 3905–3917.
- Perez-Hernandez, G., F. Paul, T. Giorgino, G. De Fabritiis, and F. Noé, 2013: Identification of slow molecular order parameters for markov model construction. *Journal of Chemical Physics*, **139** (1), 015 102.
- Prinz, J.-H., J. D. Chodera, and F. Noé, 2014: Spectral rate theory for two-state kinetics. *Physical Review X*, **4**, 011 020.
- Prinz, J.-H., et al., 2011: Markov models of molecular kinetics: Generation and validation. *Journal of Chemical Physics*, **134**, 174 105.
- Rohrdanz, M. A., W. Zheng, M. Maggioni, and C. Clementi, 2011: Determination of reaction coordinates via locally scaled diffusion map. *Journal of Chemical Physics*, **134**, 124 116.
- Sarich, M., F. Noé, and C. Schütte, 2010: On the approximation quality of markov state models. *Multiscale Modeling & Simulation*, **8** (4), 1154–1177.
- Saul, L. K. and M. I. Jordan, 1995: Boltzmann chains and hidden Markov models. *Advances in Neural Information Processing Systems*, 435–442.
- Teh, Y., M. Jordan, M. Beal, and D. Blei, 2006: Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, **101** (476), 1566–1581.
- Teh, Y. W. and M. I. Jordan, 2010: *Bayesian Nonparametrics: Principles and Practice*, chap. Hierarchical Bayesian nonparametric models with applications. Cambridge University Press, Cambridge.

Fast Newton methods for the group fused lasso

Matt Wytock

Machine Learning Dept.
Carnegie Mellon University
Pittsburgh, PA

Suvrit Sra

Machine Learning Dept.
Carnegie Mellon University
Pittsburgh, PA

J. Zico Kolter

Machine Learning Dept.
Carnegie Mellon University
Pittsburgh, PA

Abstract

We present a new algorithmic approach to the group fused lasso, a convex model that approximates a multi-dimensional signal via an approximately piecewise-constant signal. This model has found many applications in multiple change point detection, signal compression, and total variation denoising, though existing algorithms typically using first-order or alternating minimization schemes. In this paper we instead develop a specialized projected Newton method, combined with a primal active set approach, which we show to be substantially faster than existing methods. Furthermore, we present two applications that use this algorithm as a fast subroutine for a more complex outer loop: segmenting linear regression models for time series data, and color image denoising. We show that on these problems the proposed method performs very well, solving the problems faster than state-of-the-art methods and to higher accuracy.

1 Introduction

Given a multivariate signal y_1, y_2, \dots, y_T , with $y_t \in \mathbb{R}^n$, the (weighted) group fused lasso (GFL) estimator (Bleakley and Vert, 2011; Alaíz et al., 2013) attempts to find a roughly “piecewise-constant” approximation to this signal. It determines this approximation by solving the optimization problem

$$\underset{x_1, x_2, \dots, x_T}{\text{minimize}} \quad \frac{1}{2} \sum_{t=1}^T w_t \|x_t - y_t\|_2^2 + \sum_{t=1}^{T-1} \lambda_t \|x_t - x_{t+1}\|_2 \quad (1)$$

where x_1, x_2, \dots, x_T are the optimization variables, $w \in \mathbb{R}_+^T$ are weights for each time point, $\lambda \in \mathbb{R}_+^{T-1}$ are regularization parameters, and $\|\cdot\|_2$ denotes the Euclidean norm. Intuitively, the ℓ_2 norm on the *difference* between consecutive points encourages sparsity in

this difference: each difference $x_t - x_{t+1}$ will typically be either full or identically zero at the solution, i.e., the signal x will be approximately piecewise-constant. This approach generalizes the 1D total variation norm (Tibshirani et al., 2005; Barbero and Sra, 2011), which considers only univariate signals. Owing to the piecewise-constant nature of the approximate signals formed by the group fused lasso, the approach has found applications in signal compression, multiple change-point detection, and total variation denoising. Though several algorithms have been proposed to solve (1), to the best of our knowledge these have involved, at their foundation, first-order methods such as projected gradient, block coordinate descent, or splitting methods. Although such algorithms can sometimes obtain reasonable performance, they often fail to quickly find accurate solutions, especially when one wants to solve (1) to high precision as a “subroutine” (or prox-operator) in a larger algorithm (Barbero and Sra, 2011).

In this paper, we develop a fast algorithm for solving the optimization problem (1), based upon a projected Newton approach. Our method can solve group fused lasso problems to high numerical precision, often several orders of magnitude faster than existing state-of-the-art approaches. At its heart, our method involves dualizing the optimization problem (1) *twice*, in a particular manner, to eliminate the non-differentiable ℓ_2 norm and replace it by simple nonnegativity constraints; we solve the reformulated problem to high accuracy via a projected Newton approach. In order to fully exploit the sparsity of large-scale instances, we combine the above ideas with a primal active-set method that iteratively solves reduced-size problems to find the final set of non-zero differences for the original GFL problem.

Although our fast fused group lasso method is valuable in its own right, its real power comes when used as a proximal subroutine in a more complex algorithm, an operation that often needs to be solved thousands of times. With this motivation in mind, we apply our approach to two applications: segmenting linear regression models, and color total variation image denoising.

We demonstrate the power of our approach in experiments with real and synthetic data, both for the basic group fused lasso and these applications, and show substantial improvement over the state of the art.

2 A fast Newton method for the GFL

We begin by adopting slightly more compact notation, and rewrite (1) (the *primal* problem) as

$$\underset{X}{\text{minimize}} \quad \frac{1}{2} \|(X - Y)W^{1/2}\|_F^2 + \|XD\Lambda\|_{1,2} \quad (\text{P})$$

where $X, Y \in \mathbb{R}^{n \times T}$ denote the matrices

$$X = \begin{bmatrix} x_1 & \cdots & x_T \end{bmatrix}, \quad Y = \begin{bmatrix} y_1 & \cdots & y_T \end{bmatrix}; \quad (2)$$

$W := \text{diag}(w)$ and $\Lambda := \text{diag}(\lambda)$; $\|\cdot\|_F$ denotes the Frobenius norm; $\|\cdot\|_{1,2}$ denotes the mixed $\ell_{1,2}$ -norm

$$\|A\|_{1,2} := \sum_i \|a_i\|_2, \quad (3)$$

where a_i is the i th column of A ; and $D \in \mathbb{R}^{T, T-1}$ denotes the first order differencing operator

$$D = \begin{bmatrix} 1 & 0 & 0 & \cdots \\ -1 & 1 & 0 & \cdots \\ 0 & -1 & 1 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \quad (4)$$

so that XD takes the difference of the columns of X .

2.1 Dual problems

To solve (P), it is useful to look at its dual and (for our algorithm) a modified dual of this dual. To derive these problems, we transform (P) slightly by introducing the constraint $V = XD$, and corresponding dual variables $U \in \mathbb{R}^{n \times T-1}$. The Lagrangian is then given by

$$\mathcal{L}_P(X, U, V) := \frac{1}{2} \|(X - Y)W^{1/2}\|_F^2 + \|V\Lambda\|_{1,2} + \text{tr } U^T(V - XD). \quad (5)$$

Minimizing (5) analytically over X and V gives

$$X^* = Y - UD^T W^{-1}, \quad V^* = 0 \text{ iff } \|u_t\|_2 \leq \lambda_t \quad (6)$$

where u_t is the t -th column of U ; this leads to the dual

$$\begin{aligned} &\underset{U}{\text{maximize}} \quad -\frac{1}{2} \|UD^T W^{-1/2}\|_F^2 + \text{tr } UD^T Y^T \\ &\text{subject to} \quad \|u_t\|_2 \leq \lambda_t, \quad t = 1, \dots, T-1. \end{aligned} \quad (\text{D})$$

Indeed, several past algorithmic approaches have solved (D) directly using projected gradient methods, see e.g., (Alaíz et al., 2013).

The basis of our algorithm is to form the dual of (D), but in a manner that leads to a different problem than

the original primal. In particular, noting that the constraint $\|u_t\|_2 \leq \lambda_t$ is equivalent to the constraint that $\|u_t\|_2^2 \leq \lambda_t^2$, we can remove the non-differentiable ℓ_2 norm, and form the Lagrangian

$$\begin{aligned} \mathcal{L}_D(U, z) = & -\frac{1}{2} \|UD^T W^{-1/2}\|_F^2 + \text{tr } UD^T Y^T \\ & + \sum_{t=1}^{T-1} z_t (\|u_t\|_2^2 - \lambda_t^2). \end{aligned} \quad (7)$$

Minimizing over U analytically yields

$$U^* = YD(D^T W^{-1}D + Z)^{-1}, \quad (8)$$

where $Z := \text{diag}(z)$, and leads to the dual problem (the dual of the dual of (P))

$$\underset{z \geq 0}{\min} \quad \frac{1}{2} YD(D^T W^{-1}D + Z)^{-1} D^T Y^T + \frac{1}{2} (\lambda^2)^T z, \quad (\text{DD})$$

where λ^2 denotes squaring λ elementwise. This procedure, taking the dual of the dual of the original optimization problem, has transformed the original, non-smooth problem into a smooth optimization problem subject to a non-negativity constraint, a setting for which there are several efficient algorithms. Although (DD) is not easily solved via a standard form semidefinite program—it involves a matrix fractional term, for which the standard semidefinite programming form is computationally unattractive—it can be solved efficiently by a number of methods for smooth, bound-constrained optimization. However, as we will see below, the Hessian for this problem is typically poorly conditioned, so the choice of algorithm for minimizing (DD) has a large impact in practice. Furthermore, because the z dual variables are non-zero only for the change points of the original X variables, we expect that for many regimes we will have very few non-zero z values. These points motivate the use of projected Newton methods (Bertsekas, 1982), which perform Newton updates on the variables not bound ($z \neq 0$).

2.2 A projected Newton method for (DD)

Denote the objective of (DD) as $f(z)$; the gradient and Hessian of f are given by

$$\begin{aligned} \nabla_z f(z) &= -\frac{1}{2} (U^2)^T \mathbf{1} + \frac{1}{2} \lambda^2, \\ \nabla_z^2 f(z) &= U^T U \circ (D^T W^{-1}D + Z)^{-1}, \end{aligned} \quad (9)$$

where as above $U = YD(D^T W^{-1}D + Z)^{-1}$, U^2 denotes elementwise squaring of U , and \circ denotes the elementwise (Hadamard) product. The projected Newton method proceeds as follows: at each iteration, we construct the set of *bound* variables

$$\mathcal{I} := \{i : z_i = 0 \text{ and } (\nabla_z f(z))_i > 0\}. \quad (10)$$

We then perform a Newton update only on those variables that are *not* bound ($\bar{\mathcal{I}}$, referred to as the *free set*), and project back onto the feasible set

$$z_{\bar{\mathcal{I}}} \leftarrow [z_{\bar{\mathcal{I}}} - \alpha (\nabla_z^2 f(z))_{\bar{\mathcal{I}}, \bar{\mathcal{I}}}^{-1} (\nabla_z f(z))_{\bar{\mathcal{I}}}]_+, \quad (11)$$

Algorithm 1 Projected Newton for GFL

input signal $Y \in \mathbb{R}^{n \times T}$; weights $w \in \mathbb{R}_+^T$; regularization parameters $\lambda \in \mathbb{R}_+^{T-1}$; tolerance ϵ

output: optimized signal $X \in \mathbb{R}^{n \times T}$

initialization: $z \leftarrow 0$

repeat

1. Form dual variables and gradient

$$U \leftarrow YD(DW^{-1}D + Z)^{-1}$$

$$\nabla_z f(z) \leftarrow -\frac{1}{2}(U^2)^T 1 + \frac{1}{2}\lambda^2$$

2. Compute active constraints

$$\mathcal{I} \leftarrow \{i : z_i = 0 \text{ and } (\nabla_z f(z))_i > 0\}$$

3. Compute reduced Hessian and Newton direction

$$H \leftarrow U_{\bar{\mathcal{I}}}^T U_{\bar{\mathcal{I}}} \circ (D^T W^{-1} D + Z)_{\bar{\mathcal{I}}, \bar{\mathcal{I}}}^{-1}$$

$$\Delta z_{\bar{\mathcal{I}}} \leftarrow -H^{-1}(\nabla_z f(z))_{\bar{\mathcal{I}}}$$

4. Update variables

$$z_{\bar{\mathcal{I}}} \leftarrow [z_{\bar{\mathcal{I}}} + \alpha \Delta z_{\bar{\mathcal{I}}}]_+$$

where α is chosen by line search

until $\|(\nabla_z f(z))_{\bar{\mathcal{I}}}\|_2 \leq \epsilon$

where α is a step size (chosen by backtracking, interpolation, or other line search), and $[\cdot]_+$ denotes projection onto the non-negative orthant. The full method is shown in Algorithm 1. Although the projected Newton method is conceptually simple, it involves inverting several (possibly $T \times T$ matrices), which is impractical if these were to be computed as general matrix operations. Fortunately, there is a great amount of structure that can be exploited in this problem.

Efficiently solving $YD(D^T W^{-1} D + Z)^{-1}$. One key operation for the weighted GFL problem is to solve linear systems of the form $D^T W^{-1} D + Z$, where W and Z are diagonal. Fortunately, the first matrix is highly structured: it is a symmetric tridiagonal matrix

$$D^T W^{-1} D = \begin{bmatrix} \frac{1}{w_1} + \frac{1}{w_2} & -\frac{1}{w_2} & 0 & \cdots \\ -\frac{1}{w_2} & \frac{1}{w_2} + \frac{1}{w_3} & -\frac{1}{w_3} & \cdots \\ 0 & -\frac{1}{w_3} & \frac{1}{w_3} + \frac{1}{w_4} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}, \quad (12)$$

and adding Z to it only affects the diagonal. LAPACK has customized routines for solving problems of this form: `dpttrf` (which computes the LDL^T factorization of the matrix) and `dptts2` (which computes the solution to $LDL^T X = B$ via backsubstitution). For

our work, we modified this latter code slightly to solve systems with the unknown on the left hand side, as is required for our setting; this lends a slight speedup by exploiting the memory locality of column-based matrices. The methods factor $T-1 \times T-1$ matrix in $O(T)$ time, and solve n left hand sides in time $O(Tn)$.

Computing entries of $(D^T W^{-1} D + Z)^{-1}$. The projected Newton method also requires more than just solving equations of the form above: to compute the Hessian, we must actually also compute entries of the inverse $(D^T W^{-1} D + Z)^{-1}$ — we need to compute the entries with rows and columns in $\bar{\mathcal{I}}$. Naively, this would require solving $k = |\bar{\mathcal{I}}|$ left hand sides, corresponding to the unit bases for the entries in $\bar{\mathcal{I}}$; even using the fast solver above, this takes time $O(Tk)$. To speed up this operation, we instead use a fast method for computing the actual entries of the inverse of this tridiagonal, using an approach based upon (Usmani, 1994); this ultimately lets us compute the k^2 entries in $O(k^2)$ time, which can be much faster for small free sets.

Specifically, let $a \in \mathbb{R}^{T-1}$ and $b \in \mathbb{R}^{T-2}$ denote the diagonal and the negative off-diagonal entries of $D^T W^{-1} D + Z$ respectively (that is, $a_i = \frac{1}{w_i} + \frac{1}{w_{i+1}} + z_i$ and $b_i = \frac{1}{w_{i+1}}$), we can compute individual entries of $(D^T W^{-1} D + Z)^{-1}$ as follows (the following adapts the algorithm in (Usmani, 1994), but has enough simplifications for our case that we state it explicitly here). Define $\theta, \phi \in \mathbb{R}^T$ via the recursions

$$\begin{aligned} \theta_{i+1} &= a_i \theta_i - b_{i-1}^2 \theta_{i-1}, \quad i = 2, \dots, T-1 \\ \theta_1 &= 1, \quad \theta_2 = a_1, \\ \phi_i &= a_i \phi_{i+1} - b_i^2 \phi_{i+2}, \quad i = T-2, \dots, 1 \\ \phi_T &= 1, \quad \phi_{T-1} = a_{T-1}. \end{aligned} \quad (13)$$

Then, the (i, j) entry of $(D^T W^{-1} D + Z)^{-1}$ for $j \leq i$ is given by

$$(D^T W^{-1} D + Z)_{ij}^{-1} = \frac{1}{\theta_T} \left(\prod_{k=i}^{j-1} b_k \right) \theta_i \phi_{j+1}. \quad (14)$$

Finally, we can compute all the needed running products $\prod_{k=i}^{j-1} b_k$ by computing a single cumulative sum of the logs of the b_i terms $c_i = \sum_{j=1}^i \log b_i$ and then using the equality $\prod_{k=i}^{j-1} b_k = \exp(c_j - c_i)$.

2.3 A primal active set approach

Using the two optimizations mentioned above, the projected Newton method can very quickly find a solution accurate to numerical precision for medium sized problems (T and n on the order of thousands). However, for problems with substantially larger T , which are precisely those we are most interested in for many GFL applications, the approach above begins to break down. There are two reasons for this: 1) The size of the free

set $k = |\mathcal{I}|$, though often small at the final solution, can be significantly larger at intermediate iterations; since the Newton method ultimately does involve an $O(k^3)$ time to invert the Hessian restricted to the free set, this can quickly render the algorithm impractical. 2) Even with small free sets, the basic $O(Tn)$ cost required for a single pass over the data at each Newton iteration starts to dominate, especially since a significant number of iterations to find the correct free set may be required (only after finding the correct free set does one obtain quadratic convergence rates).

To overcome these problems, we consider a further layer to the algorithm, which wraps our fast projected Newton solver inside a primal active-set method. The basic intuition is that, at the optimal solution to the original GFL problem, there will typically be very few change points in the solution X^* (these correspond exactly to those z variables that are non-zero). If we knew these changes points ahead of time, we could solve a substantially reduced (weighted) GFL problem that was equivalent to the original problem. Specifically, let $\mathcal{J} \subseteq \{1, \dots, T-1\}$ denote the optimal set of change point locations for the primal problem. By the relationship of dual problems, this will be identical to the set of free variables $\tilde{\mathcal{I}}$ at the optimal solution, but since we treat these differently in the algorithmic design we use different notation. Then the original problem

$$\underset{X}{\text{minimize}} \quad \|(X - Y)W^{1/2}\|_F^2 + \|XDA\|_{1,2}, \quad (15)$$

where $X \in \mathbb{R}^{n \times T}$, is equivalent to the reduced problem

$$\underset{X'}{\text{minimize}} \quad \|(X' - Y')W'^{1/2}\|_F^2 + \|X'D'\Lambda_{\mathcal{J},\mathcal{J}}\|_{1,2} \quad (16)$$

with optimization variable $X' \in \mathbb{R}^{n \times k+1}$ for $k = |\mathcal{J}|$, where $D' \in \mathbb{R}^{k+1 \times k}$ denotes the same first order differences matrix but now over only $k+1$ -sized vectors, and where Y' and $W' = \text{diag}(w')$ are defined by

$$w'_i = \sum_{j \in \mathcal{J}'_i} w_j, \quad y'_i = \frac{1}{w'_i} \sum_{j \in \mathcal{J}'_i} w_j y_j, \quad (17)$$

where we define $\mathcal{J}'_i = \{\mathcal{J}_{i-1} + 1, \dots, \mathcal{J}_i\}$ for $i = 1, \dots, k+1$ (i.e., \mathcal{J}'_i denotes the list of indices within the i th segment, there being $k+1$ segments for k change points). Furthermore, all these terms can be computed in time $O(nk)$ via cumulative sums similar to the cumulative sum used for b above (which take $O(Tn)$ to compute once, but which thereafter only require $O(kn)$ to form the reduced problem).

To see this equivalence, note first that since X only changes at the points $|\mathcal{J}|$, it immediately holds that $\|XDA\|_{1,2} = \|X'D'\Lambda_{\mathcal{J},\mathcal{J}}\|_{1,2}$. To show that the other

term in the objective is also equivalent, we have that

$$\begin{aligned} & \|(X - Y)W^{1/2}\|_F^2 \\ &= \sum_{i=1}^{k+1} \|(x'_i 1^T - Y_{\mathcal{J}'_i})W_{\mathcal{J}'_i, \mathcal{J}'_i}^{1/2}\|_F^2 \\ &= \sum_{i=1}^{k+1} \left((w_{\mathcal{J}'_i}^T 1) x'^T_i x'_i - 2x'^T_i Y_{\mathcal{J}'_i} w_{\mathcal{J}'_i} + \|Y_{\mathcal{J}'_i} W_{\mathcal{J}'_i, \mathcal{J}'_i}^{1/2}\|_F^2 \right) \\ &= \sum_{i=1}^{k+1} w'_i \|x'_i - y'_i\|_2^2 + c. \end{aligned}$$

This equivalence motivates a primal active set method where we iteratively guess the active set \mathcal{J} (with some fixed limit on its allowable size), use the projected Newton algorithm to solve the reduced problem, and then use the updated solution to re-estimate the active set. This is essentially equivalent to a common “block pivoting” strategy for non-negative least squares (Portugal et al., 1994) or ℓ_1 methods (Lee et al., 2007), and has been shown to be very efficient in practice (Kim and Park, 2010). The full algorithm, which we refer to as Active Set Projected Newton (ASPN, pronounced “aspen”), is shown in Algorithm 2. In total, the algorithm is extremely competitive compared to past approaches to GFL, as we show in Section 4, often outperforming the existing state of the art by orders of magnitude.

3 Applications

Although the ASPN algorithm for the group fused lasso is a useful algorithm in its own right, part of the appeal of a fast solver for this type of problem is the possibility of using it as a “subroutine” within solvers for more complex problems. In this section we derive such algorithms for two instances: segmentation of time-varying linear regression models and multi-channel total variance image denoising. Both models have been considered in the literature previously, and the method presented here offers a way of solving these optimization problems to a relatively high degree of accuracy using simple methods.

3.1 Linear model segmentation

In this setting, we observe a sequence of input/output pairs $(a_t \in \mathbb{R}^n, y_t \in \mathbb{R})$ over time and the goal is to find model parameters x_t such that $y_t \approx a_t^T x_t$ (it is more common to denote the input itself as x_t and model parameters θ_t , but the notation here is more in keeping with the rest of this paper). Naturally, if x_t is allowed to vary arbitrarily, we can always find (an infinite number of) x_t ’s that fit the output perfectly, but if we constrain the sum of norms $\|x_t - x_{t-1}\|_2$, then we will instead look for piecewise constant segments in the parameter space; this model was apparently first proposed in Ohlsson et al. (2010).

Algorithm 2 Active Set Projected Newton (ASPN)

input signal $Y \in \mathbb{R}^{n \times T}$; weights $w \in \mathbb{R}_+^T$; regularization parameters $\lambda \in \mathbb{R}_+^{T-1}$; maximum active set size k_{\max} ; tolerance ϵ

output: optimized signal $X \in \mathbb{R}^{n \times T}$

initialization: $z \leftarrow 0$

repeat

1. Form dual variables and gradient

$$U \leftarrow YD(DW^{-1}D + Z)^{-1}$$
$$\nabla_z f(z) \leftarrow -\frac{1}{2}(U^2)^T 1 + \frac{1}{2}\lambda^2$$

2. Compute active set, containing all non-zero z_i 's and additional element with negative gradients, up to size k_{\max}

$$\mathcal{J}^0 \leftarrow \{i : z_i > 0\}$$
$$\mathcal{J}^1 \leftarrow \{i : z_i = 0, \nabla_z f(z) < 0\}$$
$$\mathcal{J} \leftarrow \mathcal{J}^0 \cup \mathcal{J}_{1:k_{\max}-|\mathcal{J}^0|}^1$$

3. Form reduced problem (Y', w') for \mathcal{J} using (17) and solve using projected Newton

$$z_{\mathcal{J}} \leftarrow \text{Projected-Newton}(Y', w', \lambda_{\mathcal{J}})$$

until $\|(\nabla_z f(z))_{\mathcal{J}}\|_2 \leq \epsilon$

This model may be cast as the optimization problem

$$\underset{X}{\text{minimize}} \|A \text{vec } X - y\|_2^2 + \|XD\Lambda\|_{1,2}, \quad (18)$$

where $X \in \mathbb{R}^{n \times T}$ is the same optimization variable as previously, $y \in \mathbb{R}^T$ denotes the vector of outputs, vec denotes the vectorization of a matrix (stacking its columns into a single column vector), and $A \in \mathbb{R}^{T \times Tn}$ is the block diagonal matrix

$$A = \begin{bmatrix} a_1^T & 0 & 0 & \cdots \\ 0 & a_2^T & 0 & \cdots \\ 0 & 0 & a_3^T & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}. \quad (19)$$

While this problem looks very similar to the ordinary GFL setting, the introduction of the additional matrix A renders it substantially more complex. While it is possible to adapt the Newton methods above to solve the problem directly, much of the special problem structure is lost, and it requires, for examples, forming $Tn \times Tn$ block tridiagonal matrices, which is substantially more computationally intensive, especially for large n (the methods scale like $O(n^3)$). While optimization may still be possible with such approaches, we instead adopt a different approach that builds on the

alternating direction method of multipliers (ADMM), an algorithm that has attracted great attention recently (e.g. Boyd et al. (2011)). Briefly, ADMM solves problems of the form

$$\underset{x,z}{\text{minimize}} f(x) + g(z), \quad \text{subject to } Ax + Bz = c, \quad (20)$$

via a sequence of alternating minimizations over x and z and dual variable updates.

The “standard” ADMM algorithm. The simplest way to apply ADMM to (18), considered for the pure group fused lasso e.g., in Wahlberg et al. (2012), is to introduce variables $Z = XD$, and formulate the problem as

$$\underset{X,Z}{\text{minimize}} \|A \text{vec } X - Y\|_2^2 + \|Z\Lambda\|_{1,2} \quad (21)$$

subject to $XD = Z$.

After some derivations, this leads to the updates

$$X^{k+1} \leftarrow \underset{X}{\text{argmin}} \|A \text{vec } X - y\|_2^2 + \frac{\rho}{2} \|XD - Z^k + U^k\|_F^2$$
$$Z^{k+1} \leftarrow \underset{Z}{\text{argmin}} \|Z\Lambda\|_{1,2} + \frac{\rho}{2} \|X^{k+1}D - Z + U^k\|_F^2$$
$$U^{k+1} \leftarrow U^k + X^{k+1}D - Z^{k+1}, \quad (22)$$

where ρ acts effectively like a stepsize for the problem. This set of updates is particularly appealing because minimization over X and Z can both be computed in closed form: the minimization over X is unconstrained quadratic optimization, and has the solution

$$(A^T A + \rho F^T F)^{-1} (A^T y + \rho F^T \text{vec}(Z^k - U^k)) \quad (23)$$

where $F = (D^T \otimes I)$. Furthermore, these updates can be computed very efficiently, since $(A^T A + \rho F^T F)$ is block tridiagonal, and since this matrix does not change at each iteration, we can precompute its (sparse) Cholesky decomposition once, and use it for all iterations; using these optimizations, the X update takes time $O(Tn^2)$. Similarly, the Z update is a proximal operator that can be solved by *soft thresholding* the columns of $X^{k+1}D + U^k$ (an $O(Tn)$ operation). Although these elements make the algorithm appealing, they hide a subtle issue: the matrix $(A^T A + \rho F^T F X)$ is poorly conditioned (owing to the poor conditioning of $D^T D$), even for large ρ . Because of this, ADMM needs a large number of iterations for converging to a reasonable solution; even if each iteration is quite efficient, the overall algorithm can still be impractical.

ADMM using the GFL proximal operator. Alternatively, we can derive a different ADMM algorithm by considering instead the formulation

$$\underset{X,Z}{\text{minimize}} \|A \text{vec } X - Y\|_2^2 + \|ZD\Lambda\|_{1,2} \quad (24)$$

subject to $X = Z$,

which leads to the iterative updates

$$\begin{aligned} X^{k+1} &\leftarrow \underset{X}{\operatorname{argmin}} \|A \operatorname{vec} X - y\|_2^2 + \frac{\rho}{2} \|X - Z^k + U^k\|_F^2 \\ Z^{k+1} &\leftarrow \underset{Z}{\operatorname{argmin}} \|Z D \Lambda\|_{1,2} + \frac{\rho}{2} \|X^{k+1} - Z + U^k\|_F^2 \\ U^{k+1} &\leftarrow U^k + X^{k+1} - Z^{k+1}. \end{aligned} \quad (25)$$

The X update can still be computed in closed form

$$\operatorname{vec} X^{k+1} = (A^T A + \rho I)^{-1} (A^T y + \rho \operatorname{vec}(Z^k - U^k)),$$

which is even simpler to compute than in the previous case, since $A^T A + \rho I$ is block diagonal with blocks $a_i a_i^T + \rho I$, which can be solved for in $O(n)$ time; thus the entire X update takes times $O(Tn)$. The downside is that the Z update, of course, can no longer be solved with soft-thresholding. But the Z update here is precisely in the form of the group fused lasso; thus, we can use ASPN directly to perform the Z update. The main advantage here is that the matrix $A^T A + \rho I$ is much better conditioned, which translates into many fewer iterations of ADMM. Indeed, as we show below, this approach can be many orders of magnitude faster than straight ADMM, which is already a very competitive algorithm for solving these problems.

3.2 Color total variation denoising

Next, we consider color total variation denoising example. Given an $m \times n$ RGB image represented as a third order tensor, $Y \in \mathbb{R}^{3 \times m \times n}$, total variation image denoising (Rudin et al., 1992; Blomgren and Chan, 1998) attempts to find an approximation $X \in \mathbb{R}^{3 \times m \times n}$ such that differences between pixels in X favor being zero. It does this by solving the optimization problem

$$\begin{aligned} \underset{X}{\operatorname{minimize}} \quad & \frac{1}{2} \|X - Y\|_F^2 + \lambda \sum_{i=1}^m \sum_{j=1}^{n-1} \|X_{:,i,j} - X_{:,i,j+1}\|_2 \\ & + \lambda \sum_{i=1}^{m-1} \sum_{j=1}^n \|X_{:,i,j} - X_{:,i+1,j}\|_2, \end{aligned} \quad (26)$$

corresponding to an ℓ_2 norm penalty on the difference between all adjacent pixels, where each pixel $X_{:,i,j}$ is represented as a 3 dimensional vector. We can write this as a sum of $m + n$ group fused lasso problems

$$\begin{aligned} \underset{X}{\operatorname{minimize}} \quad & \sum_{i=1}^m (\|X_{:,i,:} - Y_{:,i,:}\|_F^2 + \lambda \|X_{:,i,:} D\|_{1,2}) \\ & + \sum_{j=1}^n (\|X_{:,:,j} - Y_{:,:,j}\|_F^2 + \lambda \|X_{:,:,j} D\|_{1,2}), \end{aligned}$$

where $X_{:,i,:} \in \mathbb{R}^{3 \times n}$ denotes the slice of a single row of the image and $X_{:,:,j} \in \mathbb{R}^{3 \times m}$ denotes the slice of a single column.

Unfortunately, this optimization problem cannot be solved directly via the group fused lasso, as the difference penalties on the rows and columns for the same matrix X render the problem quite different from the basic GFL. We can, however, adopt an approach similar to the one above, and create separate variables corresponding to the row and column slices, plus a constraint that they be equal; formally, we solve

$$\begin{aligned} \underset{X,Z}{\operatorname{minimize}} \quad & \sum_{i=1}^m (\|X_{:,i,:} - Y_{:,i,:}\|_F^2 + \lambda \|X_{:,i,:} D\|_{1,2}) \\ & + \sum_{j=1}^n (\|Z_{:,:,j} - Y_{:,:,j}\|_F^2 + \lambda \|Z_{:,:,j} D\|_{1,2}) \\ \text{subject to} \quad & X = Z. \end{aligned} \quad (27)$$

The major advantage of this approach is that it decomposes the problem into $m + n$ independent GFL tasks, plus a meta-algorithm that adjusts each sub-problem to make the rows and columns agree. Several such algorithms are possible, including ADMM; we present here a slightly simpler scheme known as the “proximal Dykstra” method (Combettes and Pesquet, 2011), which has been previously applied to the case of (single channel, i.e., black and white) total variation denoising (Barbero and Sra, 2011). Starting with $X^0 = Y$, $P^0 = 0$, $Q^0 = 0$, the algorithm iterates as follows:

$$\begin{aligned} Z_{:,i,j}^{k+1} &\leftarrow \text{GFL}(X_{:,i,j}^k + P_{:,i,j}^k, \lambda), \quad j = 1, \dots, n \\ P^{k+1} &\leftarrow P^k + X^k - Z^{k+1} \\ X_{:,i,:}^{k+1} &\leftarrow \text{GFL}(Z_{:,i,:}^{k+1} + Q_{:,i,:}^k, \lambda), \quad i = 1, \dots, m \\ Q^{k+1} &\leftarrow Q^k + Z^{k+1} - X^{k+1}. \end{aligned} \quad (28)$$

Typically, very few iterations (on the order of 10) of this outer loop are need to converge to high accuracy. Furthermore, because each of the m or n GFL problems solved in the first and third steps are independent, they can be trivially parallelized.

4 Experimental results

We present experimental results for our approaches, both on the basic group fused lasso problem, where we compare to several other potential approaches, and on the two applications of linear model segmentation and color total variation denoising. C++ and MATLAB code implementing our methods is available at <http://www.cs.cmu.edu/~mwytock/gfl/>.

4.1 Group fused lasso

Here we evaluate the ASPN algorithm versus several alternatives to solving the group fused lasso problem, evaluated on both synthetic and real data. Figure 1

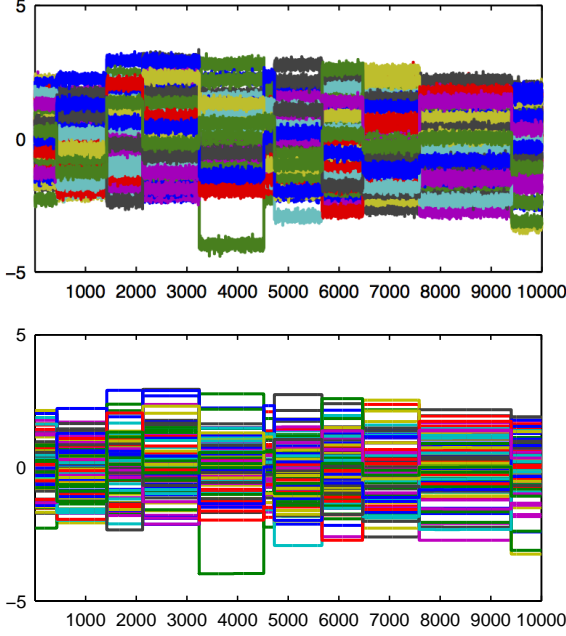


Figure 1: Above: synthetic change point data, with $T = 10000$, $n = 100$, and 10 true change points. Below: recovered signal.

shows a synthetic time series with $T = 10,000$, $n = 100$, and 10 discrete change points in the data; the data was generated by uniformly sampling the change points, sampling the mean of each segment from $\mathcal{N}(0, I)$, and then additional Gaussian noise. Figure 1 shows the recovered signal using the group fused lasso with $w_t = 1$, $\lambda_t = 20$. In Figure 2, we show timing results for this problem as well as a smaller problem with $T = 1000$ and $n = 10$; we compare ASPN to GFLseg (Bleakley and Vert, 2011) (which uses coordinate descent on the primal problem), an accelerated projected gradient on the dual (i.e., the FISTA algorithm) (Beck and Teboulle, 2009), Douglas-Rachford splitting (Combettes and Pesquet, 2007) (a generalization of ADMM that performs slightly better here), a projected gradient on the dual (Alaíz et al., 2013), and LBFGS-B (Byrd et al., 1995) applied to the dual of the dual. In all cases, ASPN performs as well as (often much better than) the alternatives.

Next, we evaluate how the ASPN algorithm scales as a function of the number of time points T and the number of change points at the solution, k . In Figure 3, the first set of experiments shows that when the number of change points at the solution is fixed ($k = 10$), the amount of time required for a highly accurate solution remains small even for large T , agreeing with analysis that shows the number of operations required is $O(T)$. In particular, a solution accurate to 10^{-6} is found in 4.8 seconds on a problem with $T = 10^6$ time points.

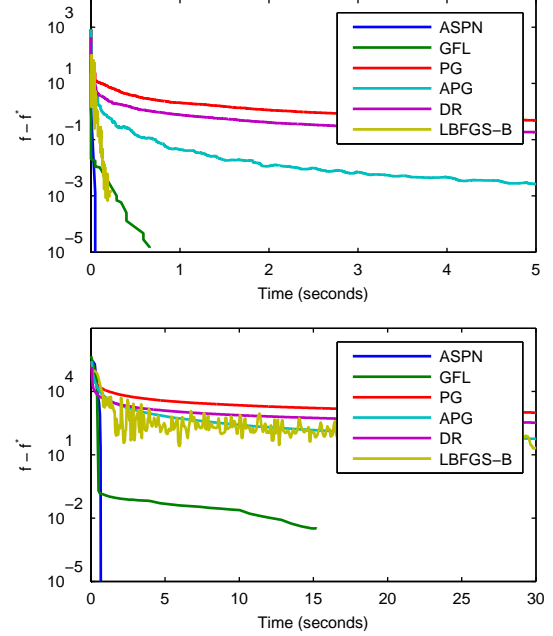


Figure 2: Above: timing results on synthetic problem with $T = 1000$, $n = 10$. Below: timing results on synthetic problem with $T = 10000$, $n = 100$.

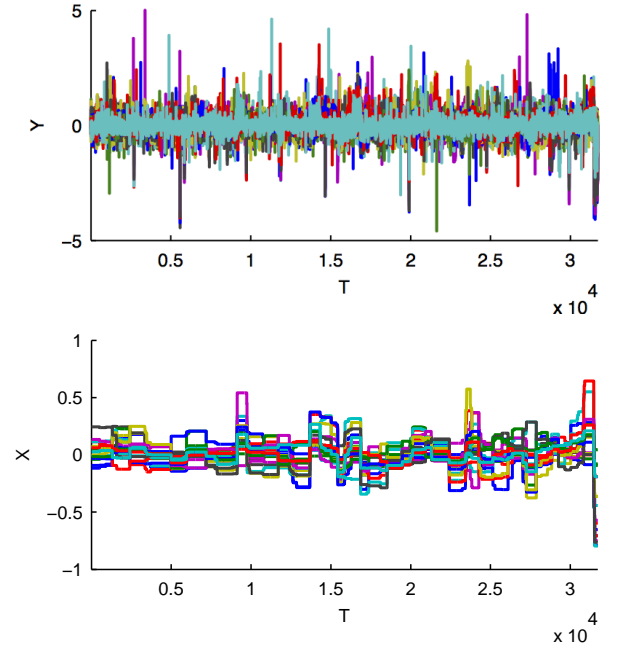


Figure 4: Above: Lung data from (Bleakley and Vert, 2011). Below: recovered signal using group fused lasso.

However, in the next set of experiments, we see that compute time grows rapidly as a function of k due to the $O(k^3)$ operations required to compute the Newton step, suggesting that the proposed method is most appropriate for problems with sparse solutions.

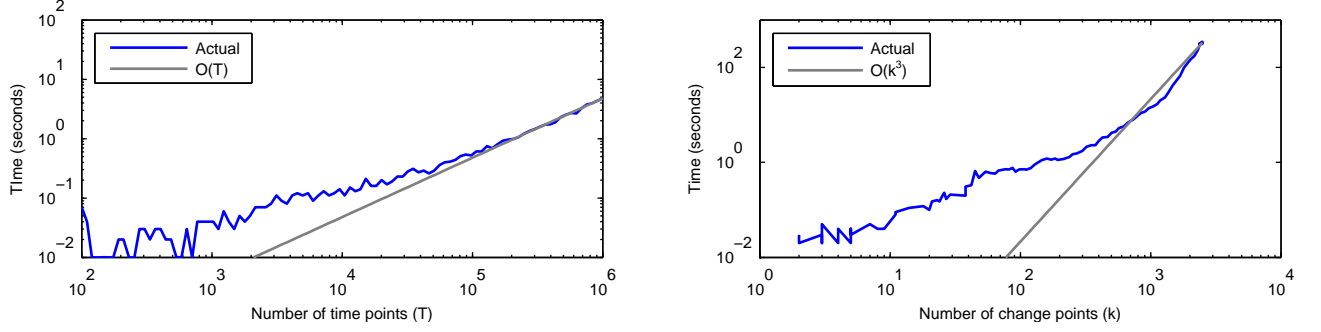


Figure 3: Left: timing results vs. number of change points at solution for synthetic problem with $T = 10000$ and $n = 10$. Right: timing results for varying T , $n = 10$, and sparse solution with 10 change points.

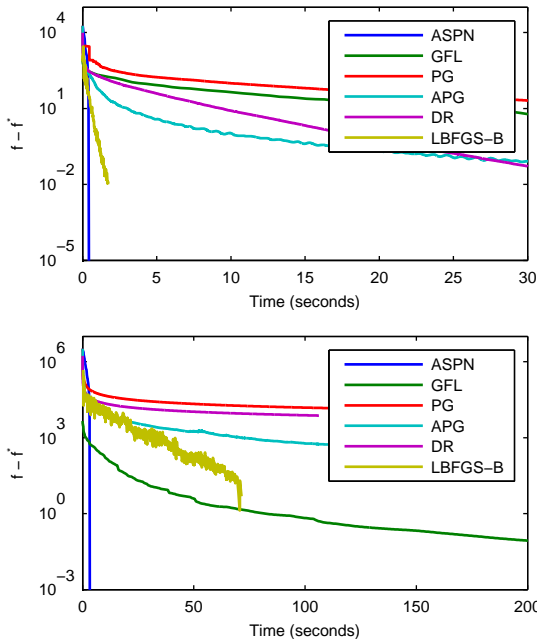


Figure 5: Above: Timing results on bladder problem, $T = 2143$, $n = 57$. Below: Timing results on lung problem, $T = 31708$, $n = 18$.

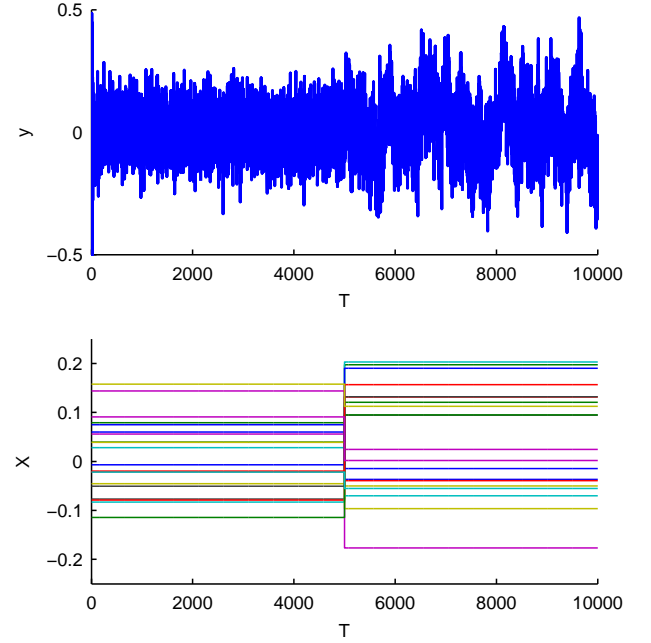


Figure 6: Above: Observed autoregressive signal z_t . Below: true autoregressive model parameters.

Finally, we evaluate the algorithm on two real time series previously used with the group fused lasso (Bleakley and Vert, 2011), from DNA profiles of bladder and lung cancer sequences. Figure 4 shows one of these two series, along with the approximation produced by the group fused lasso. Figure 5 shows timing results for the above methods again on this problem: here we observe the same overall behavior, that ASPN typically dominates the other approaches.

4.2 Linear regression segmentation

Here we apply the two different ADMM methods discussed in Section 3.1 to the task of segmenting auto-

regressive time series models. In particular, we observe some time series z_1, \dots, z_T , and we fit a linear model to this data $z_t \approx a_t^T x_t$ where $a_t = (z_{t-1}, z_{t-2}, \dots, z_{t-n})$. Figure 6 shows an example time series generated by this process, as well as the true underlying model that generated the data (with additional noise). This is the rough setting used in (Ohlsson et al., 2010), which was the first example we are aware of that uses such regularization techniques within a linear regression framework. Figure 7 shows the model parameters recovered using the method from Section 3.1, which here match the ground truth closely.

Of more importance, though, is the comparison between the two different ADMM approaches. Figure 8

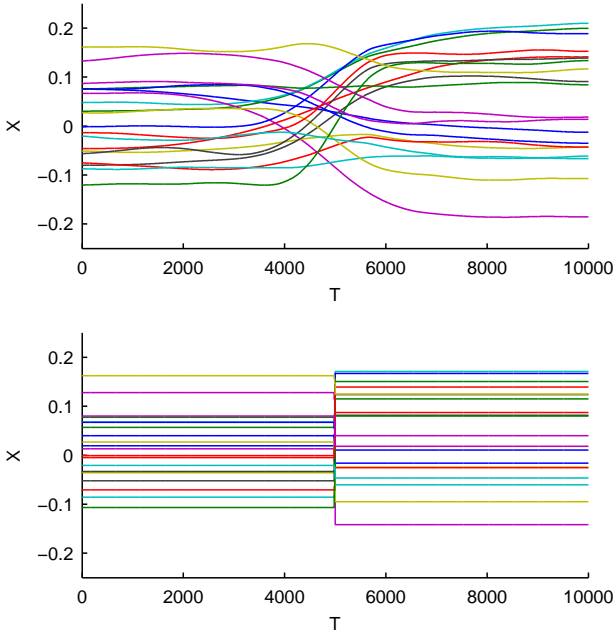


Figure 7: Above: Autoregressive parameters recovered with “simple” ADMM algorithm. Below: parameters recovered using alternative ADMM w/ ASPN.

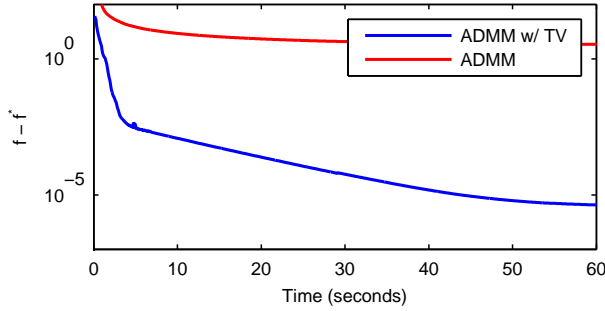


Figure 8: Convergence of simple ADMM versus alternative ADMM w/ ASPN.

shows convergence versus running time and here the “simple” ADMM approach, which encodes the difference operator in the constraints (and thus has simpler updates), converges significantly slower than our alternative. Importantly, the X axis in this figure is measured in time, and we emphasize that even though the “simple” ADMM updates are individually slightly faster (they do not involve GFL subproblems), their overall performance is much poorer. Further, as illustrated in Figure 7, the “simple” ADMM approach never actually obtains a piecewise constant X except at the optimum, which is never reached in practice.



Figure 9: Left: original image. Middle: image corrupted with Gaussian noise. Right: image recovered with total variation using proximal Dykstra and ASPN.

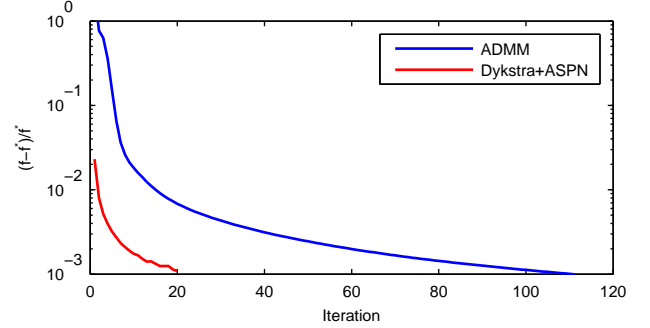


Figure 10: Comparison of proximal Dykstra method to ADMM for TV denoising of color image.

4.3 Color total variation denoising

Finally, as described in Section 3.2, we apply the proximal Dykstra algorithm, using ASPN as a fast sub-routine, to color image denoising. Figure 9 shows a 256x256 image generated by combining various solid-colored shapes, corrupted with per-RGB-component noise of $\mathcal{N}(0, 0.1)$, and then recovered with total variation denoising. There has been enormous work on total variation denoising, and while a full comparison is beyond the scope of this paper, ADMM or methods such as those used by the FTVd routines in Yang et al. (2009), for instance, are considered to be some of the fastest for this problem. In Figure 10, we show the performance of our approach and ADMM versus iteration number, and as expected observe better convergence; for single-core systems, ADMM is ultimately a better solution for this problem, since each iteration of ADMM takes about 0.767 seconds in our implementation whereas 512 calls to ASPN take 20.4 seconds. However, the advantage to the ASPN approach is that all these calls can be trivially parallelized for a 256X speedup (the calls are independent and all code is CPU-bound), whereas parallelizing a generic sparse matrix solve, as needed for ADMM-based approaches, is much more challenging and thus per-iteration performance highlights the potential benefits of the ASPN approach.

Acknowledgements This work was supported in part by the National Science Foundation under award IIS-1320402, and by support from Google.

References

- Alaíz, C. M., Jiménez, Á. B., and Dorronsoro, J. R. (2013). Group fused lasso. In *International Conference on Artificial Neural Networks and Machine Learning (ICANN)*, pages 66–73.
- Barbero, Á. and Sra, S. (2011). Fast newton-type methods for total variation regularization. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, pages 313–320.
- Beck, A. and Teboulle, M. (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202.
- Bertsekas, D. P. (1982). Projected newton methods for optimization problems with simple constraints. *SIAM Journal on control and Optimization*, 20(2):221–246.
- Bleakley, K. and Vert, J.-P. (2011). The group fused lasso for multiple change-point detection. *arXiv preprint arXiv:1106.4199*.
- Blomgren, P. and Chan, T. F. (1998). Color TV: total variation methods for restoration of vector-valued images. *Image Processing, IEEE Transactions on*, 7(3):304–309.
- Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122.
- Byrd, R. H., Lu, P., Nocedal, J., and Zhu, C. (1995). A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208.
- Combettes, P. L. and Pesquet, J.-C. (2007). A Douglas-Rachford splitting approach to nonsmooth convex variational signal recovery. *Selected Topics in Signal Processing, IEEE Journal of*, 1(4):564–574.
- Combettes, P. L. and Pesquet, J.-C. (2011). Proximal splitting methods in signal processing. In *Fixed-point algorithms for inverse problems in science and engineering*, pages 185–212. Springer.
- Kim, J. and Park, H. (2010). Fast active-set-type algorithms for L1-regularized linear regression. In *International Conference on Artificial Intelligence and Statistics*, pages 397–404.
- Lee, H., Battle, A., Raina, R., and Ng, A. Y. (2007). Efficient sparse coding algorithms. In *Neural Information Processing Systems*.
- Ohlsson, H., Ljung, L., and Boyd, S. (2010). Segmentation of ARX-models using sum-of-norms regularization. *Automatica*, 46(6):1107–1111.
- Portugal, L. F., Judice, J. J., and Vicente, L. N. (1994). A comparison of block pivoting and interior-point algorithms for linear least squares problems with nonnegative variables. *Mathematics of Computation*, 63(208):625–643.
- Rudin, L. I., Osher, S., and Fatemi, E. (1992). Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1):259–268.
- Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., and Knight, K. (2005). Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(1):91–108.
- Usmani, R. A. (1994). Inversion of a tridiagonal jacobi matrix. *Linear Algebra and Its Applications*, 212:413–414.
- Wahlberg, B., Boyd, S., Annergren, M., and Wang, Y. (2012). An ADMM algorithm for a class of total variation regularized estimation problems. *arXiv preprint arXiv:1203.1828*.
- Yang, J., Yin, W., Zhang, Y., and Wang, Y. (2009). A fast algorithm for edge-preserving variational multichannel image restoration. *SIAM Journal on Imaging Sciences*, 2(2):569–592.

Learning from Point Sets with Observational Bias

Liang Xiong

Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA 15213

Jeff Schneider

Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Many objects can be represented as sets of multi-dimensional points. A common approach to learning from these point sets is to assume that each set is an *i.i.d.* sample from an unknown underlying distribution, and then estimate the similarities between these distributions. In realistic situations, however, the point sets are often subject to sampling biases due to variable or inconsistent observation actions. These biases can fundamentally change the observed distributions of points and distort the results of learning. In this paper we propose the use of conditional divergences to correct these distortions and learn from biased point sets effectively. Our empirical study shows that the proposed method can successfully correct the biases and achieve satisfactory learning performance.

1 INTRODUCTION

Traditional learning algorithms deal with fixed, finite dimensional vectors/points, but many real objects are actually sets of points that are multi-dimensional, real-valued vectors. For instance, in computer vision an image is often treated as a set of patches with each patch described by a fixed length feature vector (Li and Perona, 2005). In monitoring problems, each sensor produces one set of measurements for a particular region within a time period. In a social network, a community is a set of people. It is important to devise algorithms that can effectively process and learn from these data.

A convenient and often adopted way to deal with point sets is to construct a feature vector for each set so that standard learning techniques can be applied. However, this conversion process often relies on human effort and domain expertise and is prone to information loss. Recently, several algorithms were proposed to directly learn from point sets

based on the assumption that each set is a sample from an underlying distribution. (Póczos et al., 2011, 2012) proposed novel kernels between point sets based on efficient and consistent divergence estimators. (Gretton et al., 2007; Muandet et al., 2012) designed a class of set kernels based on the kernel embedding of distributions. (Boiman et al., 2008; McCann and Lowe, 2012) developed simple classifiers for point sets based on divergences between the sets and the classes. Some parametric methods have also been proposed (Jaakkola and Haussler, 1998; Jebara et al., 2004). These methods achieved impressive empirical successes, thus showing the advantage of learning directly from point sets.

One factor that can significantly affect the effectiveness of learning is sampling bias. Sampling bias comes from the way we collect points from the underlying distributions, and makes the observed sample not representative of the true distribution. It undermines the fundamental validity of learning because the points are no longer iid samples from a distribution conditioned only on the object's type. Though it has been extensively studied in statistics, this key problem has been largely ignored by the previous research on learning from sets. The goal of this paper is to alleviate the impact of sampling bias when measuring similarities between point sets.

We consider point sets with the following structure. Let each point be described by two groups of random variables: the independent variables (*i.v.*) and dependent variables (*d.v.*). A point is collected by first specifying the value of the *i.v.*, and then observing a sample from the distribution of the *d.v.* conditioned on the given *i.v.* Figure 1 shows a synthetic example where the *i.v.* is sampled uniformly, and the *d.v.* is from the Gaussian distribution whose mean is proportional to the value of *i.v.*, forming the black line-shaped point set. Many real world situations, including surveys and mobile sensing, produce point sets of this type. In patch-based image analysis, we first specify the location of the patches as the *i.v.* and then extract their features as the *d.v.* In traffic monitoring, a helicopter is sent to specific locations at specific times (*i.v.*) and measures the traffic

volume ($d.v.$).

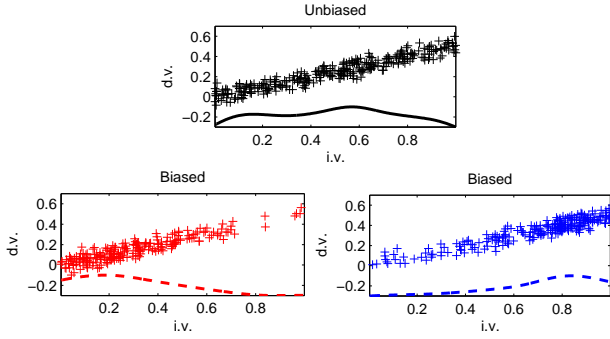


Figure 1: The observation biases.

We assume that the sampling bias affects the way we observe $i.v.$, yet the observation of $d.v.$ given $i.v.$ remains intact. This assumption is compatible with the covariate shift model (Shimodaira, 2000; Huang et al., 2007). As shown in Figure 1, an unbiased observer will sample $i.v.$ uniformly and get the black set. Biased observers might focus more on the smaller or larger values of the $i.v.$ and create the biased red and blue sets, where the curves show the observed marginal densities of the $i.v.$ The joint and marginal distributions of the biased sets now look very different from each other and the unbiased set. Nevertheless, no matter what the distribution of $i.v.$ is, the distribution of $d.v.$ given $i.v.$ is always the same Gaussian that does not change with the observer. In traffic monitoring, the helicopter may be tasked with other, non-traffic, jobs that create different patrol schedules each day, thus creating an uneven profile of the city’s traffic. But the measured traffic volumes at the patrolled locations are still accurate.

To correct sampling biases of this kind, we propose to use conditional divergences. Existing divergence-based methods use the joint distribution of the $i.v.$ and the $d.v.$ to measure the differences between point sets. On the other hand, conditional divergences focus on the conditional distributions of $d.v.$ given $i.v.$ and are insensitive to the distribution of $i.v.$, which is distorted by the sampling bias in our setting. As long as the conditional distributions are intact, the conditional divergences will be reliable. Moreover, it can be shown that the divergence between joint distributions is a special case of the conditional divergence. A fast and consistent estimator is developed for the conditional divergences. We also discuss specific examples of correcting sampling biases, including some extreme cases.

We evaluate the effectiveness of conditional divergences on both synthetic and real world data sets. On synthetic data sets, we show that the proposed estimator is accurate and the conditional divergences are capable of correcting sampling biases. We also demonstrate their performance on real-world climate and image classification problems.

The rest of this paper is organized as follows. The back-

ground and some related work is introduced in Section 2. Section 3 defines the conditional divergence and describes its properties and estimation. Section 4 describes how to use conditional divergence to correct various sampling biases. In Section 5 we make a discussion about the conditional divergences. In Section 6, we evaluate the effectiveness of the proposed methods on both synthetic and real data sets. We conclude the paper in Section 7.

2 BACKGROUND AND RELATED WORK

We consider a data set that consists of M point sets $\{G_m\}_{m=1,\dots,M}$, and each point set G_m is a set of d -dimensional vectors, $G_m = \{z_{mn}\}_{n=1,\dots,N_m}$, $z_{mn} \in \mathbb{R}^d$. Each point $z_{mn} = [x_{mn}; y_{mn}]$ is a concatenation of two shorter vectors $x_{mn} \in \mathbb{R}^{d_x}$ and $y_{mn} \in \mathbb{R}^{d_y}$ representing the independent variables $i.v.$ and the dependent variables $d.v.$ respectively. We assume that each G_m has an underlying distribution $f_m(z) = f_m(x, y)$, and the points $\{z_{mn}\}$ are *i.i.d.* samples from $f_m(z)$. f_m can be written as $f_m(z) = f_m(y|x)f_m(x)$. In the context of image classification, each G_m is an image, and x_{mn} is the location of the n th patch and y_{mn} is the feature of that patch.

We can learn from these sets by estimating the divergence between the f_m ’s as the dissimilarity between the G_m ’s. Having the dissimilarities, various problems can be solved by using similarity based learning algorithms, including *k*-nearest neighbors (KNN), *spectral clustering* (Ng et al., 2001), and *support vector machines* (SVM). In this direction, several divergence-based methods have been proposed (Boiman et al., 2008; Póczos et al., 2012; Muandet et al., 2012), and both empirical and theoretical successes were achieved.

In the presence of sampling bias that affects the distribution of $i.v.$, $f_m(x)$ is transformed into $f'_m(x)$. Consequently the observed G_m ’s represent the biased joint distribution $f'_m(z) = f_m(y|x)f'_m(x)$. Therefore naively learning from the point sets using joint distributions will lead us to the distorted f'_m ’s instead of the true f_m ’s. To correct the sampling bias, we need to either 1) modify the point sets to restore $f(z)$, or 2) use similarity measures that are insensitive to $f(x)$.

Existing correction methods often reweigh the points in the training set so that its effective distribution matches the distribution in the test set (Shimodaira, 2000; Huang et al., 2007; Cortes et al., 2008). Our proposed conditional divergences are insensitive to the biased distributions of the independent variables and thus robust against sampling biases.

Traditionally in statistics and machine learning, sampling bias is considered between the training set and the test set. In contrast, we consider problems consisting of a large

number of point sets, and our goal is to learn from the sets themselves. This extension raises many important challenges, including how to find a common basis to compare all pairs of distributions, how to deal with unobserved segments of distributions, and how to design efficient algorithms.

To our knowledge, this is first time sampling bias is addressed in the context of learning from sets of points. Algorithms such as (Póczos et al., 2011, 2012; Gretton et al., 2007; Muandet et al., 2012; Boiman et al., 2008; McCann and Lowe, 2012; Jebara et al., 2004) all directly compare the joint distributions of the observed points, making them susceptible to sample bias. (Póczos, 2012) proposed the use of conditional divergence, yet sampling bias was still not considered.

3 CONDITIONAL DIVERGENCES

We propose to measure the dissimilarity between two distributions $p(z) = p(x, y)$ and $q(z) = q(x, y)$ using the *conditional divergence* (CD) based on the *Kullback-Leibler* (KL) divergence:

$$\text{CD}_{c(x)}(p(z)||q(z)) = \mathbb{E}_{c(x)} [\text{KL}(p(y|x)||q(y|x))] \quad (1)$$

where $c(x)$ is a user-specified distribution over which the expectation is taken. CD is the average KL divergence between the conditional distributions $p(y|x)$ and $q(y|x)$ over possible values of x , and $c(x)$ can be considered as the importance of the divergences at different x 's. CD's definition is free of the i.v. distributions $p(x)$ and $q(x)$, which are vulnerable to sampling biases. By definition, CD has a lot in common with the KL divergence: it is non-negative, and equals zero if and only if $p(y|x) = q(y|x)$ for every x within the support of $c(x)$. CD is also not a metric and not even symmetric.

In the form of (1), CD is hard to compute because the divergences $\text{KL}(p(y|x)||q(y|x))$ are not available for arbitrary continuous distributions. Also note that $c(x)$ is a distribution specified by the user. To make CD more accessible, we can rewrite it as

$$\begin{aligned} \text{CD}_{c(x)}(p(z)||q(z)) &= \mathbb{E}_{p(z)} \left[\frac{c(x)}{p(x)} \left(\ln \frac{p(z)}{q(z)} - \ln \frac{p(x)}{q(x)} \right) \right]. \end{aligned} \quad (2)$$

Now, CD is defined in terms of the density ratios of the input distributions and the expectation over $p(z)$.

An interesting case of (2) occurs when we choose $c(x) = p(x)$, which gives the result

$$\begin{aligned} \text{CD}_{p(x)}(p(z)||q(z)) &= \text{KL}(p(z)||q(z)) - \text{KL}(p(x)||q(x)). \end{aligned} \quad (3)$$

We can see this special CD is equal to the *joint divergence* (divergence between joint distributions) minus the divergence between the marginal distributions of x . Intuitively, CD is removing the effect of $p(x)$ and $q(x)$ from the joint divergence, so that the net results are free from the sampling bias. Moreover, when $p(x)$ and $q(x)$ are the same, $\text{KL}(p(x)||q(x))$ vanishes and this CD equals the joint divergence. In other words, when there is no sampling bias, $\text{CD}_{p(x)}(p(z)||q(z)) = \text{KL}(p(z)||q(z))$.

3.1 ESTIMATION

In this section we give an estimator for CD (2). Suppose we have two sets G_p and G_q with underlying distributions $p(z)$ and $q(z)$ respectively. We can approximate the expectation (2) with the empirical mean and estimated densities:

$$\begin{aligned} \widehat{\text{CD}}_{c(x)}(p(z)||q(z)) &= \frac{1}{N_p} \sum_{n=1}^{N_p} \frac{c(x_{p,n})}{\hat{p}(x_{p,n})} \left(\ln \frac{\hat{p}(z_{p,n})}{\hat{q}(z_{p,n})} - \ln \frac{\hat{p}(x_{p,n})}{\hat{q}(x_{p,n})} \right), \end{aligned} \quad (4)$$

where N_p is the size of G_p , \hat{p}, \hat{q} are the estimates of p, q .

$c(t)$ is an arbitrary input from the user and we can see that its role is to reweight the log-density-ratios at different points in G_p . To generalize this notion, we define the *generalized conditional divergence* (GCD) and its estimator as the weighted average of the log-density-ratios:

$$\text{GCD}_w(p(z)||q(z)) \quad (5)$$

$$= \sum_{n=1}^{N_p} w(x_{p,n}) \left(\ln \frac{p(z_{p,n})}{q(z_{p,n})} - \ln \frac{p(x_{p,n})}{q(x_{p,n})} \right)$$

$$\widehat{\text{GCD}}_w(p(z)||q(z)) \quad (6)$$

$$= \sum_{n=1}^{N_p} w(x_{p,n}) \left(\ln \frac{\hat{p}(z_{p,n})}{\hat{q}(z_{p,n})} - \ln \frac{\hat{p}(x_{p,n})}{\hat{q}(x_{p,n})} \right)$$

$$\sum_{n=1}^{N_p} w(x_{p,n}) = 1, w(x_{p,n}) \geq 0,$$

where $w(x)$ is the weight function and the constraint $\sum_n w(x_n) = 1$ is induced by the fact that

$$\begin{aligned} \lim_{N_p \rightarrow \infty} \sum_{n=1}^{N_p} w(x_{p,n}) &= \lim_{N_p \rightarrow \infty} \frac{1}{N_p} \sum_{n=1}^{N_p} \frac{c(x_{p,n})}{p(x_{p,n})} \\ &= \mathbb{E}_{p(x)} \left[\frac{c(x)}{p(x)} \right] = \int \frac{c(x)}{p(x)} p(x) dx = 1. \end{aligned}$$

To obtain the density estimates \hat{p}, \hat{q} , we use the *k-nearest-neighbor* (KNN) based estimator (Loftsgaarden and Quesenberry, 1965). Let the $f(z)$ be the d -dimensional density function to be estimated and $Z = \{z_n\}_{n=1, \dots, N} \in \mathbb{R}^d$ be samples from $f(z)$. Then the density estimate at the point

z' is

$$\hat{f}(z') = \frac{k}{N c_1(d) \phi_{Z,k}^d(z')}, \quad (7)$$

where $c_1(d)$ is the volume of the unit ball in the d -dimensional space, and $\phi_{Z,k}(z')$ denotes the distance from z' to its k th nearest neighbor in Z (if z' is already in Z then it is excluded). This estimator is chosen over other options such as the *kernel density estimation* because it is simple, fast, and leads to a provably convergent estimator as shown below.

By plugging in (7) into (6), we can get the following estimator for GCD:

$$\begin{aligned} \widehat{\text{GCD}}_w(p(z)||q(z)) \\ = \sum_{n=1}^{N_p} w(x_{p,n}) \left(d \ln \frac{\phi_{G_q,k}(z_{p,n})}{\phi_{G_p,k}(z_{p,n})} - d_x \ln \frac{\phi_{G_q,k}(x_{p,n})}{\phi_{G_p,k}(x_{p,n})} \right), \end{aligned} \quad (8)$$

where d_x is the dimensionality of the x . We can see that the resulting estimator has a simple form and can be calculated based only on the KNN statistics ϕ , which are efficient to compute using space-dividing trees or even approximate KNN algorithms such as (Muja and Lowe, 2009). Also note that even though the estimator (8) is obtained using the density estimator (7), its final form only involves simple combinations of the log-KNN-statistics $\ln \phi$. Thus, this GCD estimator effectively avoids explicit density estimation which is notoriously difficult, especially in high dimensions.

More importantly, the GCD estimator (8) has stronger convergence properties than the density estimator from which it is derived. Standard convergence results have that the density estimator (7) is statistically consistent only if $k/n \rightarrow 0, k \rightarrow \infty$ simultaneously. However, for estimator (8) convergence can be achieved even for a fixed finite k . This means that we can always use a small k to keep the nearest neighbor search fast and still get good estimates. Specifically, following the work of (Wang et al., 2009; Póczos and Schneider, 2011), the following theorem can be proved:

Theorem 1. *Suppose the density function pairs $(p(z), q(z))$ and $(p(x), q(x))$ are both 2-regular (as defined in (Wang et al., 2009)). Also suppose that the weight function satisfies $\lim_{N_p \rightarrow \infty} w(x_{p,n}) = 0, \forall n$. Then the estimator (8) is L^2 consistent for any fixed k . That is*

$$\begin{aligned} \lim_{N_p, N_q \rightarrow \infty} \mathbb{E} \left[\widehat{\text{GCD}}_w(p(z)||q(z)) - \text{GCD}_w(p(z)||q(z)) \right]^2 \\ = 0 \end{aligned}$$

The proof of Theorem 1 is similar to what was used in (Wang et al., 2009). The condition $\lim_{N_p \rightarrow \infty} w(x_{p,n}) = 0$ ensures that the weight function does not concentrate on only a few points. We omit the detailed proof here. Note

that the convergence of GCD does not carry to CD (4) because the weight function $w(x_{p,n}) = \frac{c(x_{p,n})}{\bar{p}(x_{p,n})}$ is no longer deterministic. However, empirically we found that (4) exhibits the behavior of a consistent estimator and produces satisfactory results.

4 CHOOSING $c(x)$

To use CD, we have to choose the appropriate $c(x)$ or $w(x)$. When learning from point sets, it is preferable to use the same $c(x)$ to compute the CDs between all pairs of sets, so that they have a common basis to compare. However, this is not always necessary or possible. Even though the choice of $c(x)$ and $w(x)$ can be arbitrary, we consider 3 options below.

First, we can let $c(x) \propto 1$ so that $w(x_{p,n}) \propto p^{-1}(x_{p,n})$ to treat every value of x equally. The disadvantage is that $p^{-1}(x_{p,n})$ has to be estimated, which is error prone. We can also use $c(x) = p(x)$ and $w(x_{p,n}) \propto 1$, leading to (3). In this case, different pairs of sets can have different $c(x)$'s. When the sampling bias is small, these differences might be acceptable considering the possible errors in $w(x)$ otherwise. Thirdly, $c(x) \propto p(x)q(x)$ and $w(x_{p,n}) \propto q(x_{p,n})$ puts the focus on regions where both $p(x)$ and $q(x)$ are high. It means that we should put larger weights in dense regions and avoid scarce regions to get reliable estimates.

One caveat is that the weight function and the log-density-ratios in CD should not use the same density estimate, otherwise the estimation errors will correlate and cause systematic overestimations. Using different estimators can help decouple the errors and avoid accumulation. In practice, we use the estimator (7) with a different k .

Some extreme cases of sampling biases are when whole segments of the distribution are missing from the sample and therefore unobserved. Two sets can even have disjoint supports of x . With the CD, we can choose $c(x) \propto p(x)q(x)$ or $c(x) \propto I(p(x)q(x) > 0)$, where $I(\cdot)$ is the indicator function, and only compare two sets in their overlapping regions. The resulting quantity may not be accurate with respect to the true unbiased divergence, but it is still a valid measurement of the differences between conditional distributions. When $f(y|x)$ only weakly depends on x , this estimate can be an adequate approximation to the original divergence. If $f(y|x)$ varies drastically for different x 's without any regularity then only comparing the overlapping regions might be the best we can do.

When two sets have disjoint supports in x , no useful information can be extracted and the corresponding divergence has to be regarded as missing without further assumptions. Nevertheless, in our settings where a large number of point sets are available, it is likely that each set will share its support in x with at least some others to provide a few reliable divergence estimates. We might be able to infer the diver-

gence between disjoint sets using the idea of triangulation. We shall leave this possibility for future investigation.

5 DISCUSSION

In CD, $c(x)$ conveys prior knowledge about the importance of different x 's. It should be carefully chosen based on the data, and poor results can happen when the assumptions made in $c(x)$ are not valid. For example, $c(x) \propto 1$ assumes that all the x 's are equally important. This could be a bad assumption when the supports of two sets do not overlap, because at some x 's one of the densities will be zero, making the conditional densities $f(y|x)$ not well-defined. Similar problems might occur in regions where one of the densities is very low. Numerically the estimator can still work but usually produces poor results. In this scenario, $c(x) \propto p(x)q(x)$ suits the data better.

The CD estimator (8) relies on the KNN statistics ϕ which is the distance between nearest neighbors. Usually we use Euclidean distance to measure the difference between points and find nearest neighbors. However, the estimator does not prevent the use of other distances. In fact, (Loftsgaarden and Quesenberry, 1965) shows that alternative distances can be used and the consistency results will generally still hold. A common choice of adaptive distance measure is the *Mahalanobis distance* (Bishop, 2007), which is equivalent to applying a linear transformation to the random variables. It is even possible to learn the distance metric for ϕ in a supervised way to maximize the learning performance. We leave this possibility as future work.

The estimated conditional divergences can be used in many learning algorithms to accomplish various tasks. In this paper, we use kernel machines to classify point sets as in (Póczos et al., 2011, 2012). Having the divergence estimates, we convert them into Gaussian kernels and then use SVM for classification. When constructing kernels, all the divergences are symmetrized by taking the average $\mu(p, q) = \frac{d(p||q) + d(q||p)}{2}$. The symmetrized divergences μ are then exponentiated to get the Gaussian kernel $k(p, q) = \exp(-\gamma\mu(p, q))$ and the kernel matrix \mathbf{K} , where γ is the width parameter. Unfortunately, \mathbf{K} usually does not represent a valid Mercer kernel because the divergence is not a metric and random estimation errors exist. As a remedy, we discard the negative eigenvalues from the kernel matrix \mathbf{K} to convert it to its closest *positive semi-definite* (PSD) matrix $\tilde{\mathbf{K}}$. This $\tilde{\mathbf{K}}$ then is a valid kernel matrix and can be used in an SVM for learning.

6 EXPERIMENTS

We examine the empirical properties of the conditional divergences and their estimators. The tested divergences are listed below.

- **Full D**: Divergence between full unbiased sets as the groundtruth.
- **D**: Divergence between biased sets.
- **D-DV**: Divergence between biased sets while ignoring the *i.v.*
- **CD-P, CD-U, CD-PQ**: conditional divergences with $c(x) \propto p(x)$, $c(x) \propto 1$, $c(x) \propto p(x)q(x)$ respectively between biased sets.

Full D, D, D-DV are estimated using the KL divergence estimator proposed by (Wang et al., 2009). Unless stated otherwise, we use $k = 3$ for GCD estimation using (8), and use k values between 30 and 50 to compute the weight function.

We consider two types of sampling biases. The first type creates different $f(x)$'s for different sets, yet they still share the same support of x as the original unbiased data. Based on the first type, the second type of sampling bias is more extreme and can hide certain segments of the true distributions, and thus causes different sets to have different supports of x . We call the resulting test sets from these two sampling biases *uneven sets* and *partial sets* respectively.

In order to evaluate the quality of the bias correction by the CDs, we use controlled sampling biases in our experiments. The original point set data are collected from real problems without any sampling bias. Then we resample each set to create artificial sampling biases. By doing this, we can compare the results using the biased sets to the divergences using the unbiased data which is the groundtruth.

An SVM is used to classify the point sets using the method described in Section 5. When using the SVM, we tune the width parameter γ and the slack penalty C by 3-fold cross-validation on the training set.

6.1 SYNTHETIC DATA

6.1.1 Estimation Accuracy

We generate synthetic data to test the accuracy of the proposed conditional divergence estimators. The data set consists of 2-dimensional (one as *i.v.* and one as *d.v.*) Gaussian noise along two horizontal lines as the two classes, as shown in Figure 2 and 3. The Gaussians have fixed spherical covariance, and the mean of the blue class is slightly higher than the red class, resulting in an analytical KL divergence of 0.5. Then the *i.v.* (x axis) is resampled to create sampling bias and the red and blue curves show the resulting marginal densities $f_{\text{red}}(x)$, $f_{\text{blue}}(x)$. The task is to recover the true divergence value 0.5 from this biased sample. We vary the sample size to see the empirical convergence, and the results of 10 random runs are reported. The shortcut for this problem is to ignore the *i.v.*, but we do not let the estimators take it and force them to recover from the sampling bias.

Figure 2 shows the results on the uneven sets. As expected, the joint divergences are corrupted by the sampling bias and are far from the truth. The three CDs all converge to the true value. Figure 3 shows the results on the partial sets. The joint divergence diverges in this case. CD-P and CD-U are closer but not converging to the correct value, and the reason is that the non-overlapping supports violate the assumptions made by them. CD-PQ successfully achieved the true value. This shows the advantage of only measuring CD within the overlapping region in this example. Overall, the CDs are effective against sampling bias and the estimators converge to the true values.

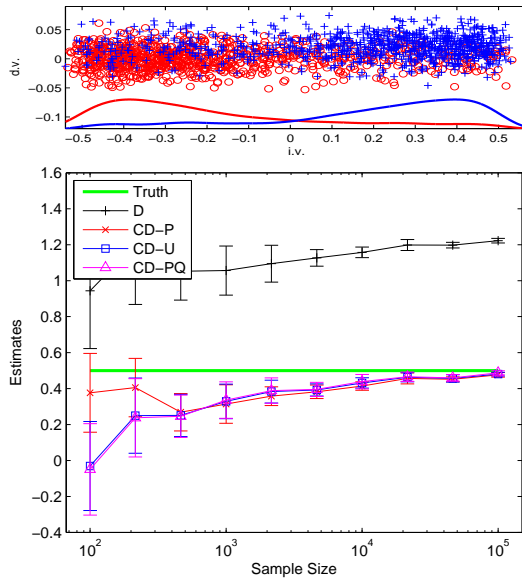


Figure 2: Divergences on the uneven synthetic data.

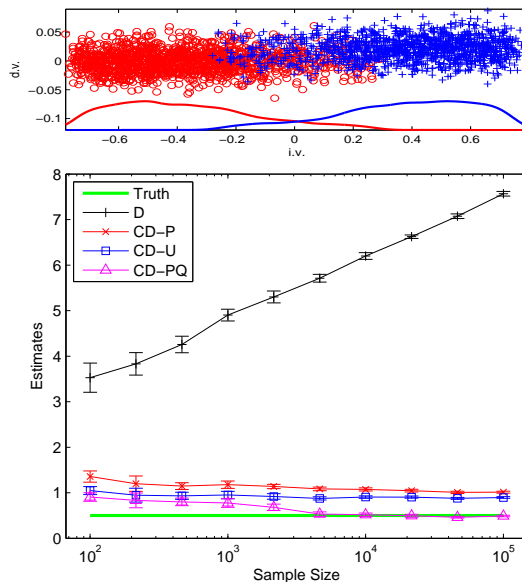


Figure 3: Divergences on the partial synthetic data.

6.1.2 Handling Point Sets

Here we test the estimators using a large number of point sets. The full data of two classes are shown in Figure 5a. To create partial sets, we use a sliding window, whose width is half of the data's span, to scan the full data and at each position put the points within the window together as a set. The uneven sets are then created by combining the partial sets with a small number of random samples from the original data. 100 sets are created for each class and each set contains 200 – 300 points.

This data set is more challenging: the marginal distribution of $d.v.$ cannot differentiate the two classes; the conditional distributions $f(y|x)$ are dependent on x ; near the center of the data the conditional distributions of the two classes are very close. The different divergence matrices on the uneven sets are shown in Figure 4, in which we sorted the sets according to their classes and window positions to show the structures. We see that the joint divergence is severely affected by the sampling bias, while the CDs are quite insensitive. The result of CD-U is especially impressive: the similarity structure of the original data is perfectly recovered. Figure 5 shows the results on the partial sets. The joint divergence is now dominated by the sampling bias. CDs again are able to recover from this severe disruption and achieve reasonable results. The result of CD-PQ is the cleanest on this data set.

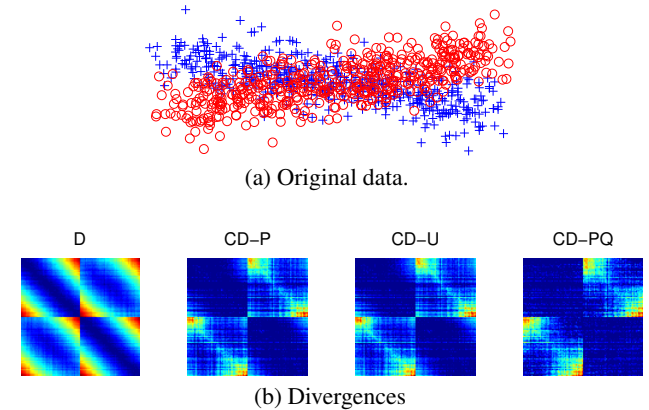


Figure 5: Divergences on the partial sets. The goal is to recover the “Full D” result shown in Figure 4.

6.2 SEASON CLASSIFICATION

In this section we use the divergences in SVM to classify real world point sets generated by sensor networks. We gathered the data from the QCLCD climate database at NCDC¹. We use a subset of QCLCD that contains daily climatological data from May 2007 to May 2013 measured by 1,164 weather stations in the continental U.S. Each of these weather station produces various measurements such

¹<http://www.ncdc.noaa.gov>

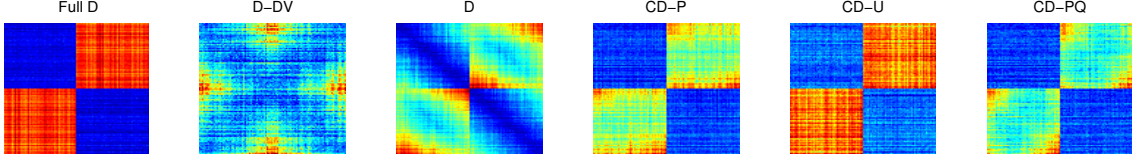


Figure 4: Divergences on the uneven sets. The goal is to recover the “Full D” given only the biased sets.

as the temperature, humidity, precipitation, *etc.*, at its location. We aggregate these data into point sets, so that each set contains the measurements from all stations in one week.

We consider the problem of predicting the season of a set based on the average temperature measurement. Specifically, we want to know if a set corresponds to Spring or Fall based on the average temperatures over the U.S. Note that classifying Summer and Winter would be too easy, while differentiating Spring and Fall can be challenging since they have similar average temperatures. Nevertheless, it is still possible based on the geographical distribution of the temperatures. Figure 6 shows the temperature maps in a first week of March and a first week of November.

Again, we create uneven and partial sets based on the original data by randomly positioning a full-width window whose height is 20% of the data’s vertical span, as shown in Figure 6. This injection of sampling bias is simulating the scenario where we only have a sensing satellite orbiting parallel to the equator. In this problem, the location is the *i.v.* and the temperature is the *d.v.*. This procedure gives us 160 3-dimensional (latitude, longitude, temperature) point sets with sizes around 2,000.

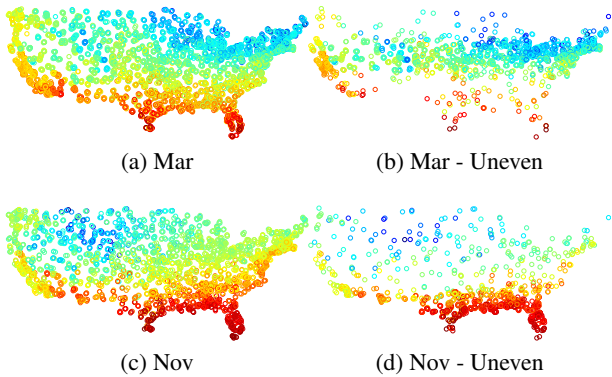
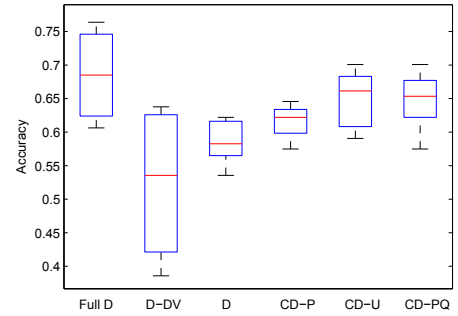


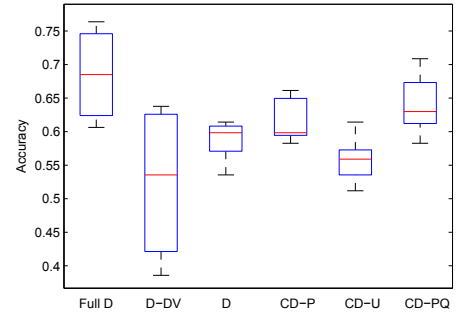
Figure 6: Example temperature maps of the U.S. from the QCLCD. (a) and (c) are the original data. (b) and (d) are the artificially created uneven data.

In each run, 20% of the random point sets are used for training and the rest are used for testing. Classification results of 10 runs are reported in Figure 7. On the uneven sets, we see that both CD-U and CD-PQ are able to recover from the sampling bias and achieve results that are only 3% worse

than the full divergence. On the partial sets, however, the performance CD-U dropped significantly. This indicates that it can be risky to apply CD in regions where two sets do not overlap. It is interesting to see that D-DV, which ignores the locations, barely does better than random since Spring and Fall indeed have similar temperatures. Yet by considering the geographical distribution of temperatures we can achieve 70% accuracy.



(a) QCLCD, uneven.



(b) QCLCD, partial.

Figure 7: Season classification results on the QCLCD weather data.

6.3 IMAGE CLASSIFICATION

We can also use CDs to classify scene images. We construct one point set for each image, where each point describes one patch including its location (*i.v.*) and the feature (*d.v.*). The OT (A.Oliva and Torralba, 2001) scene images are used, which contain 2,688 grayscale images of size 256×256 from 8 categories. The patches are sampled densely on a grid and multiscale SIFT features are extracted using VLFeat (Vedaldi and Fulkerson, 2008). The points are reduced to 20-dimensions using PCA, preserving 70%

of variance.

Again, we create both uneven and partial point sets by randomly positioning a full-width window whose height is 60% of the image. By doing this, the injected sampling bias forces a set to focus on a specific horizontal part of the scene. For instance in a beach scene, the biased observer focuses either on the sky or the sand, and only see a small part of the rest of the scene. After the above processing, the full data set contains 2,688 sets of 20-dimensional points, and the sets' sizes are around 1,600. In the biased data, each partial set has about 950 points and each uneven set has about 1,100. In each run, we randomly select 50 images per class for training and another 50 for testing.

Results of 10 random runs are shown in Figure 8. In these results, CDs again successfully restore the accuracies to a high level even in the face of harsh sampling biases. We see that CD-U impressively beats the other methods by a large margin on the uneven sets, and is only 1% worse than the full divergence. CD-PQ is the best on partial sets. These results show the CDs' corrective power when the correct assumptions are made about the sampling biases.

We also observe that CD-U and CD-P did not perform well on the partial sets, which is expected since their assumptions were invalid on the data. In general, the impact of sampling bias on this data set is small (less than 10% decrease in accuracies) because the patch features (*d.v.*) only weakly depend on the patch locations (*i.v.*). In fact, many patch-based image analyses such as (Li and Perona, 2005) do not include the locations. This might explain why both D-DV and D-P did reasonably well in this task and the corrected results by CD-PQ are only slightly better.

7 CONCLUSION

In this paper we described various aspects of dealing with sampling bias when learning from point sets. We proposed the conditional divergence (CD) to measure the difference between point sets and alleviate the impact of sampling bias. An efficient and convergent estimator of CD was provided. We then discussed how to deal with various types of sampling biases using CD. In the experiments we show that these methods are effective against sampling bias on both synthetic and real data.

Several directions can be explored in the future. We can extend the definition of conditional divergence from KL divergence to the more general Rényi divergences. The generalized conditional divergences provide the possibility of learning the weights of the density ratios in a supervised ways in order to maximize the discriminative power of the resulting divergences. The distance between points used in estimating the CDs could also be learned. Finally for extreme cases that cause missing divergences, we may infer them by exploiting the relationships among the sets using

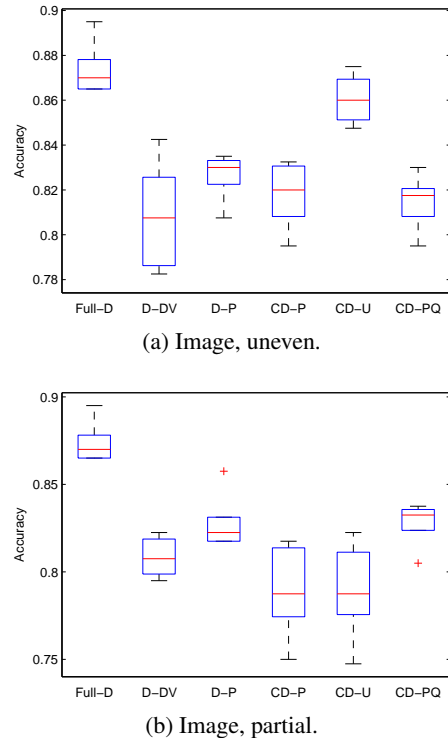


Figure 8: Image classification results on OT.

matrix completion techniques.

Acknowledgments

This research is supported by DARPA grant #FA87501220324.

References

- A.Oliva and A. Torralba. Modmodel the shape of the scene: a holistic representation of spatial envelope. *International Journal of Computer Vision (IJCV)*, 42, 2001.
- Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2007.
- Oren Boiman, Eli Shechtman, and Michal Irani. In defense of nearest neighbor based image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.
- Corinna Cortes, Mehryar Mohri, Michael Riley, and Afshin Rostamizadeh. Sample selection bias correction theory. In *Algorithmic Learning Theory*, 2008.
- Arthur Gretton, Karsten M. Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex J. Smola. A kernel method for the two sample problem. In *Neural Information Processing Systems (NIPS)*, 2007.
- Jiayuan Huang, Alexander J. Smola, Arthur Gretton, Karsten M. Borgwardt, and Bernhard Schölkopf. Cor-

- recting sample selection bias by unlabeled data. In *NIPS*, 2007.
- Tommi Jaakkola and David Haussler. Exploiting generative models in discriminative classifiers. In *NIPS*, 1998.
- T. Jebara, R. Kondor, A. Howard, K. Bennett, and N. Cesa-bianchi. Probability product kernels. *Journal of Machine Learning Research*, 5:819–844, 2004.
- Fei-Fei Li and Pietro Perona. A bayesian hierarchical model for learning natural scene categories. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2005.
- D. O. Loftsgaarden and C. P. Quesenberry. A nonparametric estimate of a multivariate density function. *The Annals of Mathematical Statistics*, 36(3), 1965.
- Sancho McCann and David G. Lowe. Local naive bayes nearest neighbor for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.
- Krikamol Muandet, Kenji Fukumizu, Francesco Dinuzzo, and Bernhard Schölkopf. Learning from distributions via support measure machines. In *Neural Information Processing Systems (NIPS)*, 2012.
- Marius Muja and David G. Lowe. Fast approximate nearest neighbor with automatic algorithms configuration. In *International Conference on Computer Vision Theory and Applications (VISAPP)*, 2009.
- Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *NIPS*, 2001.
- Barnabás Póczos. Nonparametric estimation of conditional information and divergences. In *AI and Statistics (AISTATS)*, 2012.
- Barnabás Póczos and Jeff Schneider. On the estimation of alpha divergence. In *AI and Statistics (AISTATS)*, 2011.
- Barnabás Póczos, Liang Xiong, and Jeff Schneider. Non-parametric divergence estimation with applications to machine learning on distributions. In *Uncertainty in Artificial Intelligence (UAI)*, 2011.
- Barnabás Póczos, Liang Xiong, Dougal Sutherland, and Jeff Schneider. Nonparametric kernel estimators for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.
- Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2), 2000.
- A. Vedaldi and B. Fulkerson. Vlfeat: An open and portable library of computer vision algorithms. <http://www.vlfeat.org/>, 2008.
- Qing Wang, Sanjeev R. Kulkarni, and Sergio Verdú. Divergence estimation for multidimensional densities via k-nearest-neighbor distances. *IEEE Trans. on Information Theory*, 55, 2009.

Constructing Separators and Adjustment Sets in Ancestral Graphs

Benito van der Zander, Maciej Liśkiewicz
Theoretical Computer Science
University of Lübeck, Germany
{benito, liskiewi}@tcs.uni-luebeck.de

Johannes Textor
Theoretical Biology & Bioinformatics
Utrecht University, The Netherlands
johannes.textor@gmx.de

Abstract

Ancestral graphs (AGs) are graphical causal models that can represent uncertainty about the presence of latent confounders, and can be inferred from data. Here, we present an algorithmic framework for efficiently testing, constructing, and enumerating m -separators in AGs. Moreover, we present a new constructive criterion for covariate adjustment in directed acyclic graphs (DAGs) and maximal ancestral graphs (MAGs) that characterizes adjustment sets as m -separators in a subgraph. Jointly, these results allow to find all adjustment sets that can identify a desired causal effect with multivariate exposures and outcomes in the presence of latent confounding. Our results generalize and improve upon several existing solutions for special cases of these problems.

1 INTRODUCTION

Graphical causal models endow researchers with a language to codify assumptions about a data generating process (Pearl, 2009; Elwert, 2013). Using graphical criteria, one can assess whether the assumptions encoded in such a model allow estimation of a causal effect from observational data, which is a key issue in Epidemiology (Rothman et al., 2008), the Social Sciences (Elwert, 2013) and other fields where controlled experimentation is typically impossible. Specifically, the famous back-door criterion by Pearl (2009) can identify cases where causal effect identification is possible by standard covariate adjustment, and other methods like the front-door criterion or do-calculus can even permit identification even if the back-door criterion fails (Pearl, 2009). In current practice, however, covariate adjustment is highly preferred to such alternatives because its statistical properties are well understood, giving access to useful methodology like robust estimators and confidence intervals. In contrast, knowledge about the sta-

tistical properties of e.g. front-door estimation is still considerably lacking (VanderWeele, 2009; Glynn and Kashin, 2013)¹. Unfortunately, the back-door criterion is not complete, i.e., it does not find all possible options for covariate adjustment that are allowed by a given graphical causal model.

In this paper, we aim to efficiently find a definitive answer for the following question: Given a causal graph \mathcal{G} , which covariates \mathbf{Z} do we need to adjust for to estimate the causal effect of the exposures \mathbf{X} on the outcomes \mathbf{Y} ? To our knowledge, no efficient algorithm has been shown to answer this question, not even when \mathcal{G} is a directed acyclic graph (DAG), though constructive solutions do exist for special cases like singleton $\mathbf{X} = \{X\}$ (Pearl, 2009), and a subclass of DAGs (Textor and Liśkiewicz, 2011). Here, we provide algorithms for adjustment sets in DAGs as well as in maximal ancestral graphs (MAGs), which extend DAGs allowing to account for unspecified latent variables. Our algorithms are guaranteed to find all valid adjustment sets for a given DAG or MAG with polynomial delay, and we also provide variants to list only those sets that minimize a user-supplied cost function or to quickly construct a simple adjustment set if one exists. Modelling multiple, possibly interrelated exposures \mathbf{X} is important e.g. in case-control studies that screen several putative causes of a disease (Greenland, 1994). Likewise, the presence of unspecified latent variables often cannot be excluded in real-world settings, and the causal structure between the observed variables may not be completely known. We hope that the ability to quickly deduce from a given DAG or MAG whether and how covariate adjustment can render a causal effect identifiable will benefit researchers in such areas.

We have two main contributions. First, in Section 3, we present algorithms for verifying, constructing, and listing m -separating sets in AGs. This subsumes a number of earlier solutions for special cases of these problems, e.g.

¹Quoting VanderWeele (2009), “Time will perhaps tell whether results like Pearl’s front-door path adjustment theorem and its generalizations are actually useful for epidemiologic research or whether the results are simply of theoretical interest.”

the Bayes-Ball algorithm for verification of d -separating sets (Shachter, 1998), the use of network flow calculations to find minimal d -separating sets in DAGs (Tian et al., 1998; Acid and de Campos, 2003), and an algorithm to list minimal adjustment sets for a certain subclass of DAGs (Textor and Liškiewicz, 2011). Our verification and construction algorithms for single separators are asymptotically runtime-optimal. Although we apply our algorithms only to adjustment set construction, they are likely useful in other settings as separating sets are involved in most graphical criteria for causal effect identification. Moreover, the separators themselves constitute statistically testable implications of the causal assumptions encoded in the graph.

Second, we give a graphical criterion that characterizes adjustment sets in terms of separating sets, and is sound and complete for DAGs and MAGs without selection variables. This generalizes the sound and complete criterion for DAGs by Shpitser et al. (2010), and the sound but incomplete adjustment criterion for MAGs without selection variables by Maathuis and Colombo (2013). Our criterion exhaustively addresses adjustment set construction in the presence of latent covariates and with incomplete knowledge of causal structure if at least a MAG can be specified. We give the criterion separately for DAGs (Section 4) and MAGs (Section 5) because the same graph usually admits more adjustment options if viewed as a DAG than if viewed as a MAG.

2 PRELIMINARIES

We denote sets by bold upper case letters (\mathbf{S}), and sometimes abbreviate singleton sets as $\{S\} = S$. Graphs are written calligraphically (\mathcal{G}), and variables in upper-case (X).

Mixed graphs and paths. We consider mixed graphs $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ with nodes (vertices, variables) \mathbf{V} and directed ($A \rightarrow B$), undirected ($A - B$), and bidirected ($A \leftrightarrow B$) edges \mathbf{E} . Nodes linked by an edge are *adjacent*. A *walk* of length n is a node sequence V_1, \dots, V_{n+1} such that there exists an edge sequence E_1, E_2, \dots, E_n for which every edge E_i connects V_i, V_{i+1} . Then V_1 is called the *start node* and V_{n+1} the *end node* of the walk. A *path* is a walk in which no node occurs more than once. Given a node set \mathbf{X} and a node set \mathbf{Y} , a walk from $X \in \mathbf{X}$ to $Y \in \mathbf{Y}$ is called *proper* if only its start node is in \mathbf{X} . Given a graph $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ and a node set \mathbf{V}' , the *induced subgraph* $\mathcal{G}_{\mathbf{V}'} = (\mathbf{V}', \mathbf{E}')$ contains the edges \mathbf{E}' from \mathcal{G} that are adjacent only to nodes in \mathbf{V}' .

Ancestry. A walk of the form $V_1 \rightarrow \dots \rightarrow V_n$ is *directed*, or *causal*. If there is a directed walk from U to V , then U is called an *ancestor* of V and V a *descendant* of U . A graph is *acyclic* if no directed walk from a node to itself is longer than 0. All directed walks in an acyclic graph are paths. A walk is *anterior* if it were directed after replacing all edges $U - V$ by $U \rightarrow V$. If there is an anterior path

from U to V , then U is called an *anterior* of V . All ancestors of V are anteriors of V . Every node is its own ancestor, descendant, and anterior. For a node set \mathbf{X} , the set of all of its ancestors is written as $An(\mathbf{X})$. The descendant and anterior sets $De(\mathbf{X}), Ant(\mathbf{X})$ are analogously defined. Also, we denote by $Pa(\mathbf{X}), (Ch(\mathbf{X}))$, the set of parents (children) of \mathbf{X} .

m -Separation. A node V on a walk w is called a *collider* if two arrowheads of w meet at V , e.g. if w contains $U \leftrightarrow V \leftarrow Q$. There can be no collider if w is shorter than 2. Two nodes U, V are called *collider connected* if there is a path between them on which all nodes except U and V are colliders. Adjacent vertices are collider connected. Two nodes U, V are called *m -connected* by a set \mathbf{Z} if there is a path π between them on which every node that is a collider is in $An(\mathbf{Z})$ and every node that is not a collider is not in \mathbf{Z} . Then π is called an *m -connecting path*. The same definition can be stated simpler using walks: U, V are called *m -connected* by \mathbf{Z} if there is a walk between them on which all colliders and only colliders are in \mathbf{Z} . If U, V are *m -connected* by the empty set, we simply say they are *m -connected*. If U, V are not *m -connected* by \mathbf{Z} , we say that \mathbf{Z} *m -separates* them or *blocks* all paths between them. Two node sets \mathbf{X}, \mathbf{Y} are *m -separated* by \mathbf{Z} if all their nodes are pairwise *m -separated* by \mathbf{Z} . In DAGs, *m -separation* is equivalent to the well-known *d -separation* criterion (Pearl, 2009).

Ancestral graphs and DAGs. A mixed graph $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ is called an *ancestral graph* (AG) if the following two conditions hold: (1) For each edge $A \leftarrow B$ or $A \leftrightarrow B$, A is not an ancestor of B . (2) For each edge $A - B$, there are no edges $A \leftarrow C, A \leftrightarrow C, B \leftarrow C$ or $B \leftrightarrow C$. There can be at most one edge between two nodes in an AG (Richardson and Spirtes, 2002). Syntactically, all DAGs are AGs and all AGs containing only directed edges are DAGs. An AG $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ is a *maximal ancestral graph* (MAG) if every non-adjacent pair of nodes U, V can be *m -separated* by some $\mathbf{Z} \subseteq \mathbf{V} \setminus \{U, V\}$. Every AG \mathcal{G} can be turned into a MAG \mathcal{M} by adding bidirected edges between node pairs that cannot be *m -separated* (Richardson and Spirtes, 2002).

3 ALGORITHMS FOR m -SEPARATION

In this section, we compile an algorithmic framework for solving a host of problems related to verification, construction, and enumeration of *m -separating sets* in AGs. The problems are defined in Fig. 1, which also shows the asymptotic runtime of their solutions. Throughout, n stands for the number of nodes and m for the number of edges in a graph. All of these problems except LISTSEP can be solved by rather straightforward modifications of existing algorithms (Acid and Campos, 1996; Shachter, 1998; Tian et al., 1998; Textor and Liškiewicz, 2011). We there-

fore refrain in this paper from presenting them in detail. Pseudocodes of these algorithms are shown for reference and implementation in the online version of this paper². The online version also contains proof details that had to be omitted from this paper for space reasons.

An important tool for solving similar problems for d -separation is *moralization*, by which d -separation can be reduced to a vertex cut in an undirected graph. This reduction allows to solve problems like FINDMINSEP using standard network flow algorithms (Acid and Campos, 1996). Moralization can be generalized to AGs in the following manner.

Definition 3.1 (Moralization of AGs (Richardson and Spirtes, 2002)). *Given an AG \mathcal{G} , the augmented graph $(\mathcal{G})^a$ is an undirected graph with the same node set as \mathcal{G} such that $X - Y$ is an edge in $(\mathcal{G})^a$ if and only if X and Y are collider connected in \mathcal{G} .*

Theorem 3.2 (Reduction of m -Separation to vertex cuts (Richardson and Spirtes, 2002)). *Given an AG \mathcal{G} and three node sets X, Y and Z , Z m -separates X and Y if and only if Z is an X - Y node cut in $(\mathcal{G}_{\text{Ant}(X \cup Y \cup Z)})^a$.*

A direct implementation of Definition 3.1 would lead to a suboptimal algorithm. Therefore, we first give an asymptotically optimal (linear time in output size) moralization algorithm for AGs. We then solve TESTMINSEP, FINDMINSEP, FINDMINCOSTSEP and LISTMINSEP by generalizing existing correctness proofs of the moralization approach for d -separation (Tian et al., 1998).

Not all our solutions are based on moralization, however. Moralization takes time $O(n^2)$, and TESTSEP and FINDSEP can be solved faster, i.e. in asymptotically optimal time $O(n + m)$.

Lemma 3.3 (Efficient AG moralization). *Given an AG \mathcal{G} , the augmented graph $(\mathcal{G})^a$ can be computed in time $O(n^2)$.*

Proof. The algorithm proceeds in four steps. (1) Start by setting $(\mathcal{G})^a$ to \mathcal{G} replacing all edges by undirected ones. (2) Identify all connected components in \mathcal{G} with respect to bidirected edges (two nodes are in the same such component if they are connected by a path consisting only of bidirected edges). Nodes without adjacent bidirected edges form singleton components. (3) For each pair U, V of nodes from the same component, add the edge $U - V$ to $(\mathcal{G})^a$ if it did not exist already. (4) For each component, identify all its parents (nodes U with an edge $U \rightarrow V$ where U is in the component) and link them all by undirected edges in $(\mathcal{G})^a$. Now two nodes are adjacent in $(\mathcal{G})^a$ if and only if they are collider connected in \mathcal{G} . All four steps can be performed in time $O(n^2)$. \square

Lemma 3.4. *Let X, Y, I, R be sets of nodes with $I \subseteq R$, $R \cap (X \cup Y) = \emptyset$. If there exists an m -separator Z_0 , with $I \subseteq Z_0 \subseteq R$ then $Z = \text{Ant}(X \cup Y \cup I) \cap R$ is an m -separator.*

²URL: theory.bio.uu.nl/textor/uai14.pdf

Corollary 3.5 (Ancestry of minimal separators). *Given an AG \mathcal{G} , and three sets X, Y, I , every minimal set Z over all m -separators containing I is a subset of $\text{Ant}(X \cup Y \cup I)$.*

Proof. Assume there is a minimal separator Z with $Z \not\subseteq \text{Ant}(X \cup Y \cup I)$. According to Lemma 3.4 we have that $Z' = \text{Ant}(X \cup Y \cup I) \cap Z$ is a separator with $I \subseteq Z'$. But $Z' \subseteq \text{Ant}(X \cup Y \cup I)$ and $Z' \subseteq Z$, so $Z \neq Z'$ and Z is not a minimal separator. \square

Corollary 3.5 applies to minimum-cost separators as well because every minimum-cost separator must be minimal. Now we can solve FINDMINCOSTSEP and FINDMINSIZESEP by using weighted min-cut, which takes time $O(n^3)$ using practical algorithms, and LISTMINSEP by using Takata’s algorithm to enumerate minimal vertex cuts with delay $O(n^3)$ (Takata, 2010).

However, for FINDMINSEP and TESTMINSEP, we can do better than using standard vertex cuts.

Proposition 3.6. *The task FINDMINSEP can be solved in time $O(n^2)$.*

Proof. Two algorithms are given in the online appendix, one with runtime $O(n^2)$ and one with runtime $O(nm)$. \square

Corollary 3.7. *The task TESTMINSEP can be solved in time $O(n^2)$.*

Proof. First verify whether Z is an m -separator using moralization. If not, return “no”. Otherwise, set $S = Z$ and solve FINDMINSEP. Return “yes” if the output is Z and “no”, otherwise. \square

Moralization can in the worst case quadratically increase the size of a graph. Therefore, in some cases, it may be preferable to avoid moralization if the task at hand is rather simple, as are the two tasks considered below.

Proposition 3.8. *The task FINDSEP can be solved in time $O(n + m)$.*

Proof. This follows directly from Lemma 3.4, and the fact that the set $\text{Ant}(X \cup Y \cup I) \cap R$ can be found in linear time from the MAG without moralization. Note that unlike in DAGs, two non-adjacent nodes cannot always be m -separated in ancestral graphs. \square

By modifying the Bayes-Ball algorithm (Shachter, 1998) appropriately, we get the following.

Proposition 3.9. *The task TESTSEP can be solved in time $O(n + m)$.*

Lastly, we consider the problem of listing *all* m -separators. Here is an algorithm to solve that problem with polynomial delay.

Verification: For given \mathbf{X}, \mathbf{Y} and \mathbf{Z} decide if ...		
TESTSEP	\mathbf{Z} m -separates \mathbf{X}, \mathbf{Y}	$O(n + m)$
TESTMINSEP	\mathbf{Z} m -separates \mathbf{X}, \mathbf{Y} but no $\mathbf{Z}' \subsetneq \mathbf{Z}$ does	$O(n^2)$
Construction: For given \mathbf{X}, \mathbf{Y} and auxiliary \mathbf{I}, \mathbf{R} , output ...		
FINDSEP	an m -separator \mathbf{Z} with $\mathbf{I} \subseteq \mathbf{Z} \subseteq \mathbf{R}$	$O(n + m)$
FINDMINSEP	a minimal m -separator \mathbf{Z} with $\mathbf{I} \subseteq \mathbf{Z} \subseteq \mathbf{R}$	$O(n^2)$
FINDMINCOSTSEP	a minimum-cost m -separator \mathbf{Z} with $\mathbf{I} \subseteq \mathbf{Z} \subseteq \mathbf{R}$	$O(n^3)$
Enumeration: For given $\mathbf{X}, \mathbf{Y}, \mathbf{I}, \mathbf{R}$ enumerate all ...		
LISTSEP	m -separators \mathbf{Z} with $\mathbf{I} \subseteq \mathbf{Z} \subseteq \mathbf{R}$	$O(n(n + m))$ delay
LISTMINSEP	minimal m -separators \mathbf{Z} with $\mathbf{I} \subseteq \mathbf{Z} \subseteq \mathbf{R}$	$O(n^3)$ delay

Table 1: Definitions of algorithmic tasks related to m -separation. Throughout, $\mathbf{X}, \mathbf{Y}, \mathbf{R}$ are pairwise disjoint node sets, \mathbf{Z} is disjoint with \mathbf{X}, \mathbf{Y} which are nonempty, and $\mathbf{I}, \mathbf{R}, \mathbf{Z}$ can be empty. By a minimal m -separator \mathbf{Z} , with $\mathbf{I} \subseteq \mathbf{Z} \subseteq \mathbf{R}$, we mean a set such that no proper subset \mathbf{Z}' of \mathbf{Z} , with $\mathbf{I} \subseteq \mathbf{Z}'$, m -separates the pair \mathbf{X} and \mathbf{Y} . Analogously, we define a minimal and a minimum-cost m -separator. The construction algorithms will output \perp if no set fulfilling the listed condition exists. Delay complexity for e.g. LISTMINSEP refers to the time needed to output one solution when there can be exponentially many solutions (see Takata (2010)).

```

function LISTSEP( $\mathcal{G}, \mathbf{X}, \mathbf{Y}, \mathbf{I}, \mathbf{R}$ )
  if FINDSEP( $\mathcal{G}, \mathbf{X}, \mathbf{Y}, \mathbf{I}, \mathbf{R}$ )  $\neq \perp$  then
    if  $\mathbf{I} = \mathbf{R}$  then Output  $\mathbf{I}$ 
    else
       $V \leftarrow$  an arbitrary node of  $\mathbf{R} \setminus \mathbf{I}$ 
      LISTSEP( $\mathcal{G}, \mathbf{X}, \mathbf{Y}, \mathbf{I} \cup \{V\}, \mathbf{R}$ )
      LISTSEP( $\mathcal{G}, \mathbf{X}, \mathbf{Y}, \mathbf{I}, \mathbf{R} \setminus \{V\}$ )

```

Figure 1: ListSep

Proposition 3.10. *The task LISTSEP can be solved with polynomial delay $O(n(n + m))$.*

Proof. Algorithm LISTSEP performs backtracking to enumerate all \mathbf{Z} with $\mathbf{I} \subseteq \mathbf{Z} \subseteq \mathbf{R}$ aborting branches that will not find a valid separator. Since every leaf will output a separator, the tree height is at most n and the existence check needs $O(n + m)$, the delay time is $O(n(n + m))$. The algorithm generates every separator exactly once: if initially $\mathbf{I} \subsetneq \mathbf{R}$, with $V \in \mathbf{R} \setminus \mathbf{I}$, then the first recursive call returns all separators \mathbf{Z} with $V \in \mathbf{Z}$ and the second call returns all \mathbf{Z}' with $V \notin \mathbf{Z}'$. Thus the generated separators are pairwise disjoint. This is a modification of the enumeration algorithm for minimal vertex separators (Takata, 2010). \square

4 ADJUSTMENT IN DAGS

In this section, we leverage the algorithmic framework of the last section together with a new constructive, sound and complete criterion for covariate adjustment in DAGs to solve all problems listed in Table 1 for adjustment sets instead of m -separators in the same asymptotic time. First, however, we need to introduce some more notation pertaining to the causal interpretation DAGs.

Do-operator and adjustment sets. A DAG \mathcal{G} encodes the factorization of joint distribution π for the set of vari-

ables $\mathbf{V} = \{X_1, \dots, X_n\}$ as $p(\mathbf{v}) = \prod_{j=1}^n p(x_j | pa_j)$, where pa_j denotes a particular realization of the parent variables of X_j in \mathcal{G} . When interpreted causally, an edge $X_i \rightarrow X_j$ is taken to represent a direct causal effect of X_i on X_j . For disjoint $\mathbf{X}, \mathbf{Y} \subseteq \mathbf{V}$, the (total) causal effect of \mathbf{X} on \mathbf{Y} is $p(\mathbf{y} | do(\mathbf{x}))$ where $do(\mathbf{x})$ represents an intervention that sets $\mathbf{X} = \mathbf{x}$. In a DAG, this intervention corresponds to removing all edges into \mathbf{X} , disconnecting \mathbf{X} from its parents. We denote the resulting graph as $\mathcal{G}_{\bar{\mathbf{X}}}$. Given DAG \mathcal{G} and a joint probability density π for \mathbf{V} the post-intervention distribution can be expressed in a truncated factorization formula:

$$p(\mathbf{v} | do(\mathbf{x})) = \begin{cases} \prod_{X_j \in \mathbf{V} \setminus \mathbf{X}} p(x_j | pa_j) & \text{for } \mathbf{V} \text{ consistent with } \mathbf{x} \\ 0 & \text{otherwise.} \end{cases}$$

Definition 4.1 (Adjustment (Pearl, 2009)). *Given a DAG $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ and pairwise disjoint $\mathbf{X}, \mathbf{Y}, \mathbf{Z} \subseteq \mathbf{V}$, \mathbf{Z} is called covariate adjustment for estimating the causal effect of \mathbf{X} on \mathbf{Y} , or simply adjustment, if for every distribution p consistent with \mathcal{G} we have $p(\mathbf{y} | do(\mathbf{x})) = \sum_{\mathbf{z}} p(\mathbf{y} | \mathbf{x}, \mathbf{z})p(\mathbf{z})$.*

Definition 4.2 (Adjustment criterion (Shpitser et al., 2010; Shpitser, 2012)³). *Let $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ be a DAG, and $\mathbf{X}, \mathbf{Y}, \mathbf{Z} \subseteq \mathbf{V}$ be pairwise disjoint subsets of variables. The set \mathbf{Z} satisfies the adjustment criterion relative to (\mathbf{X}, \mathbf{Y}) in \mathcal{G} if*

- (a) *no element in \mathbf{Z} is a descendant in \mathcal{G} of any $W \in \mathbf{V} \setminus \mathbf{X}$ which lies on a proper causal path from \mathbf{X} to \mathbf{Y} and*
- (b) *all proper non-causal paths in \mathcal{G} from \mathbf{X} to \mathbf{Y} are blocked by \mathbf{Z} .*

Analogously to $\mathcal{G}_{\bar{\mathbf{X}}}$, by $\mathcal{G}_{\bar{\mathbf{X}}}$ we denote a DAG obtained from \mathcal{G} by removing all edges leaving \mathbf{X} .

³In Shpitser et al. (2010), the criterion is stated using $\mathcal{G}_{\bar{\mathbf{X}}}$ instead of \mathcal{G} . However, one can easily prove that both criteria are equivalent.

4.1 CONSTRUCTIVE BACK-DOOR CRITERION

Definition 4.3 (Proper back-door graph). Let $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ be a DAG, and $\mathbf{X}, \mathbf{Y} \subseteq \mathbf{V}$ be pairwise disjoint subsets of variables. The proper back-door graph, denoted as $\mathcal{G}_{\mathbf{XY}}^{pbd}$, is obtained from \mathcal{G} by removing the first edge of every proper causal path from \mathbf{X} to \mathbf{Y} .

Note the difference between the back-door graph $\mathcal{G}_{\mathbf{X}}$ and the proper back-door graph $\mathcal{G}_{\mathbf{XY}}^{pbd}$: in $\mathcal{G}_{\mathbf{X}}$ all edges leaving \mathbf{X} are removed while in $\mathcal{G}_{\mathbf{XY}}^{pbd}$ only those that lie on a proper causal path. However, to construct $\mathcal{G}_{\mathbf{XY}}^{pbd}$ still only elementary operations are sufficient. Indeed, we remove all edges $X \rightarrow D$ in \mathbf{E} such that $X \in \mathbf{X}$ and D is in the subset, which we call $PCP(\mathbf{X}, \mathbf{Y})$, obtained as follows:

$$PCP(\mathbf{X}, \mathbf{Y}) = (De_{\overline{\mathbf{X}}}(\mathbf{X}) \setminus \mathbf{X}) \cap An_{\mathbf{X}}(\mathbf{Y}) \quad (1)$$

where $De_{\overline{\mathbf{X}}}(\mathbf{W})$ denotes descendants of \mathbf{W} in $\mathcal{G}_{\overline{\mathbf{X}}}$. $An_{\mathbf{X}}(\mathbf{W})$ is defined analogously for $\mathcal{G}_{\mathbf{X}}$. Hence, the proper back-door graph can be constructed from \mathcal{G} in linear time $O(m + n)$.

Now we propose the following adjustment criterion. For short, we will denote the set $De(PCP(\mathbf{X}, \mathbf{Y}))$ as $Dpcp(\mathbf{X}, \mathbf{Y})$.

Definition 4.4 (Constructive back-door criterion (CBC)). Let $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ be a DAG, and let $\mathbf{X}, \mathbf{Y}, \mathbf{Z} \subseteq \mathbf{V}$ be pairwise disjoint subsets of variables. The set \mathbf{Z} satisfies the constructive back-door criterion relative to (\mathbf{X}, \mathbf{Y}) in \mathcal{G} if

- (a) $\mathbf{Z} \subseteq \mathbf{V} \setminus Dpcp(\mathbf{X}, \mathbf{Y})$ and
- (b) \mathbf{Z} d-separates \mathbf{X} and \mathbf{Y} in the proper back-door graph $\mathcal{G}_{\mathbf{XY}}^{pbd}$.

Theorem 4.5. The constructive back-door criterion is equivalent to the adjustment criterion.

Proof. First observe that the conditions (a) of both criteria are identical. Assume conditions (a) and (b) of the adjustment criterion hold. We show that (b) of the constructive back-door criterion follows. Let π be any proper path from \mathbf{X} to \mathbf{Y} in $\mathcal{G}_{\mathbf{XY}}^{pbd}$. Because $\mathcal{G}_{\mathbf{XY}}^{pbd}$ does not contain causal paths from \mathbf{X} to \mathbf{Y} , π is not causal and has to be blocked by \mathbf{Z} in \mathcal{G} by the assumption. Since removing edges cannot open paths, π is blocked by \mathbf{Z} in $\mathcal{G}_{\mathbf{XY}}^{pbd}$ as well.

Now we show that (a) and (b) of the constructive back-door criterion together imply (b) of the adjustment criterion. If that were not the case, then there could exist a proper non-causal path π from \mathbf{X} to \mathbf{Y} that is blocked in $\mathcal{G}_{\mathbf{XY}}^{pbd}$ but open in \mathcal{G} . There can be two reasons why π is blocked in $\mathcal{G}_{\mathbf{XY}}^{pbd}$: (1) The path starts with an edge $X \rightarrow D$ that does not exist in $\mathcal{G}_{\mathbf{XY}}^{pbd}$. Then we have $D \in PCP(\mathbf{X}, \mathbf{Y})$. For π to be non-causal, it would have to contain a collider $C \in An(\mathbf{Z}) \cap De(D) \subseteq An(\mathbf{Z}) \cap Dpcp(\mathbf{X}, \mathbf{Y})$. But because of (a), $An(\mathbf{Z}) \cap Dpcp(\mathbf{X}, \mathbf{Y})$ is empty. (2) A collider C on π is an ancestor

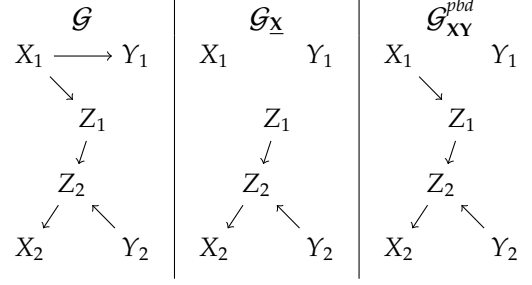


Figure 2: A DAG where for $\mathbf{X} = \{X_1, X_2\}$ and $\mathbf{Y} = \{Y_1, Y_2\}$, $\mathbf{Z} = \{Z_1, Z_2\}$ is a valid and minimal adjustment, but no set fulfills the back-door criterion (Pearl, 2009), and the parents of \mathbf{X} are not a valid adjustment set either.

of \mathbf{Z} in \mathcal{G} , but not in $\mathcal{G}_{\mathbf{XY}}^{pbd}$. Then there must be a directed path from C to \mathbf{Z} via an edge $X \rightarrow D$ with $D \in An(\mathbf{Z}) \cap PCP(\mathbf{X}, \mathbf{Y})$, contradicting (a). \square

4.2 ADJUSTING FOR MULTIPLE EXPOSURES

For a singleton set $\mathbf{X} = \{X\}$ of exposures we know that if a set of variables \mathbf{Y} is disjoint from $\{X\} \cup Pa(X)$ then one obtains easily an adjustment set with respect to X and \mathbf{Y} as $\mathbf{Z} = Pa(X)$ (Pearl, 2009, Theorem 3.2.2). The situation changes drastically if the effect of multiple exposures is estimated. Theorem 3.2.5 in Pearl (2009) claims that the expression for $P(\mathbf{y} \mid do(\mathbf{x}))$ is obtained by adjusting for $Pa(\mathbf{X})$ if \mathbf{Y} is disjoint from $\mathbf{X} \cup Pa(\mathbf{X})$, but, as the DAG in Fig. 2 shows, this is not true: the set $\mathbf{Z} = Pa(X_1, X_2) = \{Z_2\}$ is not an adjustment set according to $\{X_1, X_2\}$ and \mathbf{Y} . In this case one can identify the causal effect by adjusting for $\mathbf{Z} = \{Z_1, Z_2\}$ only. Indeed, for more than one exposure, no adjustment set may exist at all even without latent covariates and even though $\mathbf{Y} \cap (\mathbf{X} \cup Pa(\mathbf{X})) = \emptyset$, e.g. in the DAG

$$X_1 \xrightarrow{\quad} X_2 \xleftrightarrow{\quad} Z \longleftarrow Y.$$

Using our criterion, we can construct a simple adjustment set explicitly if one exists. For a DAG $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ we define the set

$$Adj(\mathbf{X}, \mathbf{Y}) = An(\mathbf{X}, \mathbf{Y}) \setminus (\mathbf{X} \cup \mathbf{Y} \cup Dpcp(\mathbf{X}, \mathbf{Y})).$$

Theorem 4.6. Let $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ be a DAG and let $\mathbf{X}, \mathbf{Y} \subseteq \mathbf{V}$ be distinct node sets. Then the following statements are equivalent:

1. There exists an adjustment in \mathcal{G} w.r.t. \mathbf{X} and \mathbf{Y} .
2. $Adj(\mathbf{X}, \mathbf{Y})$ is an adjustment w.r.t. \mathbf{X} and \mathbf{Y} .
3. $Adj(\mathbf{X}, \mathbf{Y})$ d-separates \mathbf{X} and \mathbf{Y} in the proper back-door graph $\mathcal{G}_{\mathbf{XY}}^{pbd}$.

Proof. The implication (3) \Rightarrow (2) follows directly from the criterion Def. 4.4 and the definition of $Adj(\mathbf{X}, \mathbf{Y})$. Since

the implication (2) \Rightarrow (1) is obvious, it remains to prove (1) \Rightarrow (3).

Assume there exists an adjustment set Z_0 w.r.t. X and Y . From Theorem 4.5 we know that $Z_0 \cap Dpcp(X, Y) = \emptyset$ and that Z_0 d -separates X and Y in \mathcal{G}_{XY}^{pbd} . Our task is to show that $Adj(X, Y)$ d -separates X and Y in \mathcal{G}_{XY}^{pbd} . This follows from Lemma 3.4 used for the proper back-door graph \mathcal{G}_{XY}^{pbd} if we take $I = \emptyset$, $R = V \setminus (X \cup Y \cup Dpcp(X, Y))$. \square

From Equation 1 and the definition $Dpcp(X, Y) = De(PCP(X, Y))$ we then obtain immediately:

Corollary 4.7. *Given two distinct sets $X, Y \subseteq V$, $Adj(X, Y)$ can be found in $O(n + m)$ time.*

4.3 TESTING, COMPUTING, AND ENUMERATING ADJUSTMENT SETS

Using our criterion, every algorithm for m -separating sets Z between X and Y can be used for adjustment sets with respect to X and Y , by requiring that Z not contain any node in $Dpcp(X, Y)$. This allows solving all problems listed in Table 1 for adjustment sets in DAGs instead of m -separators. Below, we name those problems analogously as for m -separation, e.g. the problem to decide whether Z is an adjustment set w.r.t. X, Y is named TESTADJ in analogy to TESTSEP.

TESTADJ can be solved by testing if $Z \cap Dpcp(X, Y) = \emptyset$ and Z is a d -separator in the proper back-door graph \mathcal{G}_{XY}^{pbd} . Since \mathcal{G}_{XY}^{pbd} can be constructed from \mathcal{G} in linear time, the total time complexity of this algorithm is $O(n + m)$.

TESTMINADJ can be solved with an algorithm that iteratively removes nodes from Z and tests if the resulting set remains an adjustment set w.r.t. X and Y . This can be done in time $O(n(n + m))$. Alternatively, one can construct the proper back-door graph \mathcal{G}_{XY}^{pbd} from \mathcal{G} and test if Z is a minimal d -separator, with $Z \subseteq V \setminus Dpcp(X, Y)$ between X and Y . This can be computed in time $O(n^2)$. The correctness of these algorithms follows from the proposition below, which is a generalization of the result in Tian et al. (1998).

Proposition 4.8. *If no single node Z can be removed from an adjustment set Z such that the resulting set $Z' = Z \setminus Z$ is no longer an adjustment set, then Z is minimal.*

The remaining problems like FINDADJ, FINDMINADJ etc. can be solved using corresponding algorithms for finding, resp. listing m -separations applied for proper back-door graphs. Since the proper back-door graph can be constructed in linear time the time complexities to solve the problems above are as listed in Table 1.

5 ADJUSTMENT IN MAGS

We now generalize the results from the previous section to MAGs. Two examples may illustrate why this generalization is not trivial. First, take $\mathcal{G} = X \rightarrow Y$. If \mathcal{G} is interpreted as a DAG, then the empty set is valid for adjustment. If \mathcal{G} is however taken as a MAG, then there exists no adjustment set as \mathcal{G} represents among others the DAG $U \rightarrow X \rightarrow Y$ where U is an unobserved confounder. Second, take $\mathcal{G} = A \rightarrow X \rightarrow Y$. In that case, the empty set is an adjustment set regardless of whether \mathcal{G} is interpreted as a DAG or a MAG. The reasons will become clear as we move on. First, let us recall the semantics of a MAG. The following definition can easily be given for AGs in general, but we do not need this generality for our purpose.

Definition 5.1 (DAG representation by MAGs (Richardson and Spirtes, 2002)). *Let $\mathcal{G} = (V, E)$ be a DAG, and let $S, L \subseteq V$. The MAG $\mathcal{M} = \mathcal{G}_{[S]}^L$ is a graph with nodes $V \setminus (S \cup L)$ and defined as follows. (1) Two nodes U and V are adjacent in $\mathcal{G}_{[S]}^L$ if they cannot be m -separated by any Z with $S \subseteq Z \subseteq V \setminus L$ in \mathcal{G} . (2) The edge between U and V is*

$$U - V \text{ if } U \in An(S \cup V) \text{ and } V \in An(S \cup U);$$

$$U \rightarrow V \text{ if } U \in An(S \cup V) \text{ and } V \notin An(S \cup U);$$

$$U \leftrightarrow V \text{ if } U \notin An(S \cup V) \text{ and } V \notin An(S \cup U).$$

We call L latent variables and S selection variables. We say there is selection bias if $S \neq \emptyset$.

Hence, every MAG represents an infinite set of underlying DAGs that all share the same ancestral relationships. For a given MAG \mathcal{M} , we can construct a represented DAG \mathcal{G} by replacing every edge $X - Y$ by a path $X \rightarrow S \leftarrow Y$, and every edge $X \leftrightarrow Y$ by $X \leftarrow L \rightarrow Y$, where S and L are new nodes; then $\mathcal{M} = \mathcal{G}_{[S]}^L$ where S and L are all new nodes. \mathcal{G} is called the *canonical DAG* of \mathcal{M} (Richardson and Spirtes, 2002), which we write as $C(\mathcal{M})$.

Lemma 5.2 (Preservation of separating sets (Richardson and Spirtes, 2002)). *Z m -separates X, Y in $\mathcal{G}_{[S]}^L$ if and only if $Z \cup S$ m -separates X, Y in \mathcal{G} .*

We now extend the concept of adjustment to MAGs in the usual way (Maathuis and Colombo, 2013).

Definition 5.3 (Adjustment in MAGs). *Given a MAG $\mathcal{M} = (V, E)$ and two variable sets $X, Y \subseteq V$, $Z \subseteq V$ is an adjustment set for X, Y in \mathcal{M} if for every probability distribution $p(\mathbf{v}')$ consistent with a DAG $\mathcal{G} = (V', E')$ for which $\mathcal{G}_{[S]}^L = \mathcal{M}$, for some $S, L \subseteq V' \setminus V$, we have*

$$p(y \mid do(x)) = \sum_z p(y \mid x, z, s)p(z \mid s). \quad (2)$$

Selection bias (i.e., $\mathbf{S} \neq \emptyset$) substantially complicates adjustment, and in fact nonparametric causal inference in general (Zhang, 2008)⁴. Due to these limitations, we restrict ourselves to the case $\mathbf{S} = \emptyset$ in the rest of this section. Note however that recovery from selection bias is sometimes possible with additional population data, and graphical conditions exist to identify such cases (Barenboim et al., 2014).

5.1 ADJUSTMENT AMENABILITY

In this section we first identify a class of MAGs in which adjustment is impossible because of causal ambiguities – e.g., the simple MAG $X \rightarrow Y$ falls into this class, but the larger MAG $A \rightarrow X \rightarrow Y$ does not.

Definition 5.4 (Visible edge (Zhang, 2008)). *Given a MAG $\mathcal{M} = (\mathbf{V}, \mathbf{E})$, an edge $X \rightarrow Y \in \mathbf{E}$ is called visible if in all DAGs $\mathcal{G} = (\mathbf{V}', \mathbf{E}')$ with $\mathcal{G}_{\mathbf{L}}^{\mathbf{L}} = \mathcal{M}$ for some $\mathbf{S}, \mathbf{L} \subseteq \mathbf{V}'$, all d -connected walks between X and Y in \mathcal{G} that contain only nodes of $\mathbf{S} \cup \mathbf{L} \cup X \cup Y$ are directed paths.*

Intuitively, an invisible directed edge $X \rightarrow Y$ means that there may still hidden confounding factors between X and Y , which is guaranteed not to be the case if the edge is visible.

Lemma 5.5 (Graphical conditions for edge visibility (Zhang, 2008)). *In a MAG $\mathcal{M} = (\mathbf{V}, \mathbf{E})$, an edge $X \rightarrow Y$ is visible if and only if there is a node A not adjacent to Y where (1) $A \rightarrow X \in \mathbf{E}$ or $A \leftrightarrow X \in \mathbf{E}$, or (2) there is a collider path $A \leftrightarrow V_1 \leftrightarrow \dots \leftrightarrow V_n \leftrightarrow X$ or $A \rightarrow V_1 \leftrightarrow \dots \leftrightarrow V_n \leftrightarrow X$ where all V_i are parents of Y .*

Definition 5.6. *We call a MAG $\mathcal{M} = (\mathbf{V}, \mathbf{E})$ adjustment amenable w.r.t. $\mathbf{X}, \mathbf{Y} \subseteq \mathbf{V}$ if all proper causal paths from \mathbf{X} to \mathbf{Y} start with a visible directed edge.*

Lemma 5.7. *If a MAG $\mathcal{M} = (\mathbf{V}, \mathbf{E})$ is not adjustment amenable w.r.t. $\mathbf{X}, \mathbf{Y} \subseteq \mathbf{V}$ then there exists no adjustment set \mathbf{W} for \mathbf{X}, \mathbf{Y} in \mathcal{M} .*

Proof. If the first edge $X \rightarrow D$ on some causal path to \mathbf{Y} in \mathcal{M} is not visible, then there exists a consistent DAG \mathcal{G} where there is a non-causal path between X and \mathbf{Y} via V that could only be blocked in \mathcal{M} by conditioning on D or some of its descendants. But such conditioning would violate the adjustment criterion in \mathcal{G} . \square

5.2 ADJUSTMENT CRITERION FOR MAGS

We now show that DAG adjustment criterion generalizes to adjustment amenable MAGs. The adjustment criterion and

⁴A counterexample is the graph $A \leftarrow X \rightarrow Y$, where we can safely assume that A is the ancestor of a selection variable. A sufficient and necessary condition for adjustment under selection bias is $\mathbf{Y} \perp\!\!\!\perp \mathbf{S} \mid \mathbf{X}$ (Barenboim et al., 2014), which is so restrictive that most statisticians would probably not even speak of “selection bias” anymore in such a case.

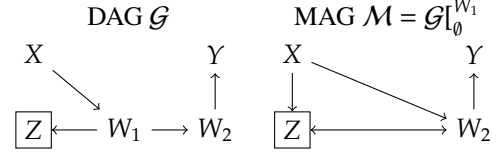


Figure 3: Illustration of the case in the proof of Theorem 5.8 where Z descends from W_1 which in a DAG \mathcal{G} is on a proper causal path from X to Y , but is not a descendant of a node on a proper causal path from X to Y in the MAG \mathcal{M} after marginalizing W_1 . In such cases, conditioning on Z will m -connect X and Y in \mathcal{M} via a proper non-causal path.

the constructive back-door criterion are defined like their DAG counterparts (Definitions 4.2 and 4.3), replacing d - with m -separation for the latter.

Theorem 5.8. *Given an adjustment amenable MAG $\mathcal{M} = (\mathbf{V}, \mathbf{E})$ and three disjoint node sets $\mathbf{X}, \mathbf{Y}, \mathbf{Z} \subseteq \mathbf{V}$, the following statements are equivalent:*

- (i) \mathbf{Z} is an adjustment relative to \mathbf{X}, \mathbf{Y} in \mathcal{M} .
- (ii) \mathbf{Z} fulfills the adjustment criterion (AC) w.r.t. (\mathbf{X}, \mathbf{Y}) in \mathcal{M} .
- (iii) \mathbf{Z} fulfills the constructive backdoor criterion (CBC) w.r.t. (\mathbf{X}, \mathbf{Y}) in \mathcal{M} .

Proof. The equivalence of (ii) and (iii) is established by observing that the proof of Theorem 4.5 generalizes to m -separation. Below we establish equivalence of (i) and (ii).

$\neg(ii) \Rightarrow \neg(i)$: If \mathbf{Z} violates the adjustment criterion in \mathcal{M} , it does so in the canonical DAG $\mathcal{C}(\mathcal{M})$, and thus is not an adjustment in \mathcal{M} .

$\neg(i) \Rightarrow \neg(ii)$: Let \mathcal{G} be a DAG with $\mathcal{G}_{\mathbf{L}_0}^{\mathbf{L}_0} = \mathcal{M}$ in which \mathbf{Z} violates the AC. We show that (a) if $\mathbf{Z} \cap Dpcp(\mathbf{X}, \mathbf{Y}) \neq \emptyset$ in \mathcal{G} then $\mathbf{Z} \cap Dpcp(\mathbf{X}, \mathbf{Y}) \neq \emptyset$ in \mathcal{M} as well, or there exists a proper non-causal path in \mathcal{M} that cannot be m -separated; and (b) if $\mathbf{Z} \cap Dpcp(\mathbf{X}, \mathbf{Y}) = \emptyset$ in \mathcal{G} and \mathbf{Z} d -connects a proper non-causal walk in \mathcal{G} , then it m -connects a proper non-causal walk in \mathcal{M} .

(a) Suppose that in \mathcal{G} , \mathbf{Z} contains a node Z in $Dpcp(\mathbf{X}, \mathbf{Y})$, and let $\mathbf{W} = PCP(\mathbf{X}, \mathbf{Y}) \cap An(\mathbf{Z})$. If \mathcal{M} still contains at least one node $W_1 \in \mathbf{W}$, then W_1 lies on a proper causal path in \mathcal{M} and Z is a descendant of W_1 in \mathcal{M} . Otherwise, \mathcal{M} must contain a node $W_2 \in PCP_{\mathcal{G}}(\mathbf{X}, \mathbf{Y}) \setminus An(\mathbf{Z})$ (possibly $W_2 \in \mathbf{Y}$) such that $W_2 \leftrightarrow A$, $X \rightarrow W_2$, and $X \rightarrow A$ are edges in \mathcal{M} , where $A \in An(\mathbf{Z})$ (possibly $A = Z$; see Fig. 3). Then \mathcal{M} contains an m -connected proper non-causal path $X \rightarrow A \leftrightarrow W \rightarrow W_2 \rightarrow \dots \rightarrow \mathbf{Y}$.

(b) Suppose that in \mathcal{G} , $\mathbf{Z} \cap Dpcp(\mathbf{X}, \mathbf{Y}) = \emptyset$, and there exists an open proper non-causal path from \mathbf{X} to \mathbf{Y} . Then there

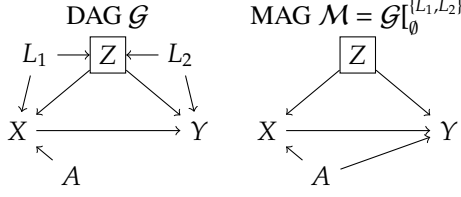


Figure 4: Case (b) in the proof of Theorem 5.8: A proper non-causal path $w_G = X \leftarrow L_1 \rightarrow Z \leftarrow L_2 \rightarrow Y$ in a DAG is d -connected by Z , but the corresponding proper non-causal path $w_M = X \leftarrow Z \rightarrow Y$ is not m -connected in the MAG, and its m -connected subpath $w'_M = X \rightarrow Y$ is proper causal. However, this also renders the edge $X \rightarrow Y$ invisible, because otherwise A could be m -separated from Y by $U = \{X, Z\}$ in M but not in G .

must then also be a proper non-causal walk w_G from some $X \in \mathbf{X}$ to some $Y \in \mathbf{Y}$ (Lemma 7.1), which is d -connected by Z in G . Let w_M denote the subsequence of w_G formed by nodes in M , which includes all colliders on w_G . The sequence w_M is a path in M , but is not necessarily m -connected by Z ; all colliders on w_M are in Z because every non- Z must be a parent of at least one of its neighbours, but there can subsequences U, Z_1, \dots, Z_k, V on w_M where all $Z_i \in Z$ but some of the Z_i are not colliders on w_M . However, then we can form from w_M an m -connected walk by bypassing some sequences of Z -nodes (Lemma 7.2). Let w'_M be the resulting walk.

If w'_M is a proper non-causal walk, then there must also exist a proper non-causal path in M (Lemma 7.1), violating the AC. It therefore remains to show that w'_M is not a proper causal path. This must be the case if w_G does not contain colliders, because then the first edge of $w_M = w'_M$ cannot be a visible directed edge out of X . Otherwise, the only way for w'_M to be proper causal is if all Z -nodes in w_M have been bypassed in w'_M by edges pointing away from X . In that case, one can show by several case distinctions that the first edge $X \rightarrow D$ of w'_M , where $D \notin Z$, cannot be visible (see Figure 4 for an example of such a case). \square

5.3 ADJUSTMENT SET CONSTRUCTION

In the previous section, we have already shown that the CBC is equivalent to the AC for MAGs as well; hence, adjustment sets for a given MAG M can be found by forming the proper back-door graph M_{XY}^{pbd} and then applying the algorithms from the previous section. In principle, care must be taken when removing edges from MAGs as the result might not be a MAG; however, this is not the case when removing only directed edges.

Lemma 5.9 (Closure of maximality under removal of directed edges). *Given a MAG M , every graph M' formed by removing only directed edges from M is also a MAG.*

Proof. Suppose the converse, i.e. M is no longer a MAG after removal of some edge $X \rightarrow D$. Then X and D cannot be m -separated even after the edge is removed because X and D are collider connected via a path whose nodes are all ancestors of X or D (Richardson and Spirtes, 2002). The last edge on this path must be $C \leftrightarrow D$ or $C \leftarrow D$, hence $C \notin An(D)$, and thus we must have $C \in An(X)$. But then we get $C \in An(D)$ in M via the edge $X \rightarrow V$, a contradiction. \square

Corollary 5.10. *For every MAG M , the proper back-door graph M_{XY}^{pbd} is also a MAG.*

For MAGs that are not adjustment amenable, the CBC might falsely indicate that an adjustment set exists even though that set may not be valid for some represented graph. Fortunately, adjustment amenability is easily tested using the graphical criteria of Lemma 5.5. For each child D of X in $Dpcp(X, Y)$, we can test the visibility of all edges $X \rightarrow D$ simultaneously using depth first search. This means that we can check all potentially problematic edges in time $O(n + m)$. If all tests pass, we are licensed to apply the CBC, as shown above. Hence, we can solve all algorithmic tasks in Table 1 for MAGs in the same way as for DAGs after an $O(k(n + m))$ check of adjustment amenability, where $k \leq |Ch(X)|$.

6 DISCUSSION

We have compiled efficient algorithms for solving several tasks related to m -separators in ancestral graphs, and applied those together with a new, constructive adjustment criterion to provide a complete and informative answer to the question when, and how, a desired causal effect can be estimated by covariate adjustment. Our results fully generalize to MAGs in the absence of selection bias. One may argue that the MAG result is more useful for exploratory applications (inferring a graph from data) than confirmatory ones (drawing a graph based on theory), as researchers will prefer drawing DAGs instead of MAGs due to the easier causal interpretation of the former. Nevertheless, in such settings the results can provide a means to construct more “robust” adjustment sets: If there are several options for covariate adjustment in a DAG, then one can by interpreting the same graph as a MAG possibly generate an adjustment set that is provably valid for a much larger class of DAGs. This might partially address the typical criticism that complete knowledge of the causal structure is unrealistic.

Our adjustment criterion generalizes the work of Shpitser et al. (2010) to MAGs and therefore now completely characterizes when causal effects are estimable by covariate adjustment in the presence of unmeasured confounders with multivariate exposures and outcomes. This also generalizes recent work by Maathuis and Colombo (2013) who provide a criterion which, for DAGs and MAGs without selection bias, is stronger than the back-door criterion but

weaker than ours. They moreover show their criterion to hold also for CPDAGs and PAGs, which represent equivalence classes of DAGs and MAGs as they are constructed by causal discovery algorithms. It is possible that the constructive back-door criterion could be generalized further to those cases, which we leave for future work.

7 APPENDIX

In this appendix, we prove Lemma 3.4 and two auxiliary Lemmas that are used in the proof of Theorem 5.8.

Proof of Lemma 3.4. Let us consider a proper walk $w = X, V_1, \dots, V_n, Y$ with $X \in \mathbf{X}, Y \in \mathbf{Y}$. If w does not contain a collider, all nodes V_i are in $\text{Ant}(\mathbf{X} \cup \mathbf{Y})$ and the walk is blocked by \mathbf{Z} , unless $\{V_1, \dots, V_n\} \cap \mathbf{R} = \emptyset$ in which case the walk is not blocked by \mathbf{Z}_0 either. If the walk contains colliders \mathbf{C} , it is blocked, unless $\mathbf{C} \subseteq \mathbf{Z} \subseteq \mathbf{R}$. Then all nodes V_i are in $\text{Ant}(\mathbf{X} \cup \mathbf{Y} \cup \mathbf{I})$ and the walk is blocked, unless $\{V_1, \dots, V_n\} \cap \mathbf{R} = \mathbf{C}$. Since $\mathbf{C} \subseteq \mathbf{Z}$ is a set of antecedents, there exists a shortest (possible containing 0 edges) path $\pi_j = V_j \rightarrow \dots \rightarrow W_j$ for each $V_j \in \mathbf{C}$ with $W_j \in \mathbf{X} \cup \mathbf{Y} \cup \mathbf{I}$ (it cannot contain an undirected edge, since there is an arrow pointing to V_j). Let $\pi'_j = V_j \rightarrow \dots \rightarrow W'_j$ be the shortest subpath of π_j that is not blocked by \mathbf{Z}_0 . Let w' be the walk w after replacing each V_j by the walk $V_j \rightarrow \dots \rightarrow W'_j \leftarrow \dots \leftarrow V_j$. If any of the W_j is in $\mathbf{X} \cup \mathbf{Y}$ we truncate the walk, such that we get the shortest walk between nodes of \mathbf{X} and \mathbf{Y} . Since π'_j is not blocked, w' contains no colliders except w'_j and all other nodes of w' are not in \mathbf{R} , w' is not blocked and \mathbf{Z}_0 is not a separator. \square

Lemma 7.1. *Given a DAG \mathcal{G} and sets $\mathbf{X}, \mathbf{Y}, \mathbf{Z} \subseteq \mathbf{V}$ satisfying $\mathbf{Z} \cap \text{Dpcp}(\mathbf{X}, \mathbf{Y}) = \emptyset$, \mathbf{Z} m -connects a proper non-causal path between \mathbf{X} and \mathbf{Y} if and only if it m -connects a proper non-causal walk between \mathbf{X} and \mathbf{Y} .*

Proof. \Leftarrow : Let w be the m -connected proper non-causal walk. It can be transformed to an m -connected path π by removing loops of nodes that are visited multiple times. Since no nodes have been added, π remains proper, and the first edges of π and w are the same. So if w does not start with a \rightarrow edge, π is non-causal. If w starts with an edge $X \rightarrow D$, there exists a collider with a descendant in \mathbf{Z} which is in $\text{De}(D)$. So π has to be non-causal, or it would contradict $\mathbf{Z} \cap \text{Dpcp}(\mathbf{X}, \mathbf{Y}) = \emptyset$.

\Rightarrow : Let π be an m -connected proper non-causal path. It can be changed to an m -connected walk w by inserting $C_i \rightarrow \dots \rightarrow Z_i \leftarrow \dots \leftarrow C_i$ for every collider C_i on π and a corresponding $Z_i \in \mathbf{Z}$. Since no edges are removed from π , w is non-causal, but not necessarily proper, since the inserted walks might contain nodes of \mathbf{X} . However, in that case, w can be truncated to a proper walk w' starting at

the last node of \mathbf{X} on w . Then w' is non-causal, since it contains the subpath $\mathbf{X} \leftarrow \dots \leftarrow C_i$. \square

Lemma 7.2. *Let $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ be a DAG and let $w_{\mathcal{G}}$ be a walk from $X \in \mathbf{V}$ to $Y \in \mathbf{V}$ that is d -connected by $\mathbf{Z} \subseteq \mathbf{V}$. Let $\mathcal{M} = \mathcal{G}_{\mathbf{L}}^{\mathbf{L}}$, where $\mathbf{L} \subseteq \mathbf{V} \setminus (\mathbf{Z} \cup \mathbf{X} \cup \mathbf{Y})$. Let $w_{\mathcal{M}} = V_1, \dots, V_{n+1}$ be the subsequence of $w_{\mathcal{G}}$ consisting only of the nodes in \mathcal{M} . Then \mathbf{Z} m -connects X and Y in \mathcal{M} via a path along a subsequence $w'_{\mathcal{G}}$ formed from $w_{\mathcal{G}}$ by removing some nodes of \mathbf{Z} (possibly $w'_{\mathcal{G}} = w_{\mathcal{G}}$).*

Proof. (Sketch) The subsequences removed from $w_{\mathcal{M}}$ correspond to maximally long inducing walks in $w_{\mathcal{G}}$ with respect to \mathbf{L} . An inducing walk is a collider connected path on which all nodes are ancestors of one of the endpoints, and all non-colliders are in \mathbf{L} . The endpoints of inducing walks with respect to \mathbf{L} must be adjacent to each other in \mathcal{M} (similar to Richardson and Spirtes, 2002, for inducing paths); It is easy to see that all \mathbf{Z} -nodes which are not colliders on $w_{\mathcal{M}}$ can be removed in this way, e.g. $X \leftarrow Z_1 \leftrightarrow Z_2 \rightarrow Y$ can be truncated to X, Y because there must have been an inducing walk in \mathcal{G} via Z_1, Z_2 . Additionally, it can be necessary to remove nodes that are colliders on $w_{\mathcal{M}}$, e.g. if $w_{\mathcal{M}} = X \leftarrow Z_1 \rightarrow Z_2 \leftarrow Y$ and $Z_2 \in \text{An}(X)$, then $w_{\mathcal{G}}$ must have been an inducing walk, and $w'_{\mathcal{M}}$ contains only X and Y even though Z_2 is a collider. To obtain the Lemma, it remains to be shown that no new colliders are created when bypassing nodes in this way. This is done by case distinctions; e.g., in the example $w_{\mathcal{M}} = X \leftarrow Z_1 \rightarrow Z_2 \leftarrow Y$ and $Z_2 \in \text{An}(X)$, we also have $Y \in \text{An}(X)$ and hence $w'_{\mathcal{M}}$ cannot be $X \rightarrow Y$ or $x \leftrightarrow y$. \square

References

- Silvia Acid and Luis M. De Campos. An algorithm for finding minimum d-separating sets in belief networks. In *Proceedings of UAI 1996*, pages 3–10, 1996.
- Silvia Acid and Luis M. de Campos. Searching for bayesian network structures in the space of restricted acyclic partially directed graphs. *J. Artif. Intell. Res. (JAIR)*, 18:445–490, 2003.
- Elias Barenboim, Jin Tian, and Judea Pearl. Recovering from selection bias in causal and statistical inference. In *Proceedings of AAAI-14*, 2014.
- Felix Elwert. *Graphical Causal Models*, pages 245–273. Handbooks of Sociology and Social Research. Springer, 2013.
- Adam Glynn and Konstantin Kashin. Front-door versus back-door adjustment with unmeasured confounding: Bias formulas for front-door and hybrid adjustments. Technical report, Harvard University, 2013.
- Sander Greenland. Hierarchical regression for epidemiologic analyses of multiple exposures. *Environ. Health Perspect.*, 102 Suppl 8:33–39, Nov 1994.
- Marloes H. Maathuis and Diego Colombo. A generalized backdoor criterion. arXiv:1307.5636, 2013.
- Judea Pearl. *Causality*. Cambridge University Press, 2009. ISBN 0-521-77362-8.
- Thomas Richardson and Peter Spirtes. Ancestral graph markov models. *Annals of Statistics*, 30:927–1223, 2002.
- Kenneth J. Rothman, Sander Greenland, and Timothy L. Lash. *Modern Epidemiology*. Wolters Kluwer, 2008. ISBN 0781755646.
- Ross D. Shachter. Bayes-ball: The rational pastime. In *Proceedings of UAI 1998*, pages 480–487, 1998.
- Ilya Shpitser. Appendix to on the validity of covariate adjustment for estimating causal effects, 2012. unpublished manuscript.
- Ilya Shpitser, Tyler VanderWeele, and James Robins. On the validity of covariate adjustment for estimating causal effects. In *Proceedings of UAI 2010*, pages 527–536. AUAI Press, 2010.
- Ken Takata. Space-optimal, backtracking algorithms to list the minimal vertex separators of a graph. *Discrete Applied Mathematics*, 158:1660–1667, 2010.
- Johannes Textor and Maciej Liśkiewicz. Adjustment criteria in causal diagrams: An algorithmic perspective. In *Proceedings of UAI*, pages 681–688, 2011.
- Jin Tian, Azaria Paz, and Judea Pearl. Finding minimal d-separators. Technical Report R-254, University of California, Los Angeles, 1998. URL ftp.cs.ucla.edu/pub/stat_ser/r254.pdf.
- Tyler J. VanderWeele. On the relative nature of overadjustment and unnecessary adjustment. *Epidemiology*, 20(4): 496–499, Jul 2009.
- Jiji Zhang. Causal reasoning with ancestral graphs. *Journal of Machine Learning Research*, 9:1437–1474, 2008.

Belief-Kinematics Jeffrey's Rules in the Theory of Evidence

Chunlai Zhou

Computer Science Department
Renmin University
Beijing, China 100872
chunlai.zhou@gmail.com

Mingyue Wang

Mathematics Department
Syracuse University
Syracuse, NY 13244-1150 USA
wangmingyue1994@gmail.com

Biao Qin

Computer Science Department
Renmin University
Beijing, China 100872
qinbiao@ruc.edu.cn

Abstract

This paper studies the problem of revising beliefs using uncertain evidence in a framework where beliefs are represented by a belief function. We introduce two *new* Jeffrey's rules for the revision based on two forms of *belief kinematics*, an evidence-theoretic counterpart of probability kinematics. Furthermore, we provide two distance measures for belief functions and show that the two belief kinematics are optimal in the sense that they minimize their corresponding distance measures.

1 INTRODUCTION

Reasoning about uncertainty is a fundamental issue for Artificial Intelligence [HALPERN, 2005]. Numerous approaches have been proposed, including Dempster-Shafer theory of belief functions [SHAFFER, 1976] (also called the theory of evidence or simply *DS* theory). Ever since the pioneering works by Dempster and Shafer, the theory of belief functions has become a powerful formalism in Artificial Intelligence for knowledge representation and decision-making.

In this paper, we study the revision of beliefs using uncertain evidence and we represent beliefs as belief functions. Our main contribution is to introduce two *new* Jeffrey's rules for the revision based on two different forms of *belief kinematics*, an evidence-theoretic counterpart of probability kinematics [JEFFREY, 1983]. The first rule, called *inner* revision, generalizes the geometric conditionalization rule, and the other, *outer* revision, generalizes the Dempster rule of conditioning. These two Jeffrey's rules specify uncertain evidence in terms of the *effect* it has on beliefs once accepted, and the specification is actually a function of both evidence strength and beliefs held prior to obtaining evidence. Once new evidence is accepted, a prior belief function *bel* on a frame Ω of discernment is revised to a

new posterior belief function *bel'* on the same frame. This method requires us to specify *uncertain* evidence by providing a belief function *bel_e* on a coarser frame with less distinctions of the attention. This coarser frame is actually based on a partition of the frame Ω and hence is represented as a subalgebra \mathcal{B} of the powerset 2^Ω of Ω whose atoms forming a partition of Ω .

The principle of belief kinematics on \mathcal{B} says that, although the prior belief function *bel* and the posterior one *bel'* may disagree on propositions in \mathcal{B} , they agree on their *relevance* to all propositions in 2^Ω . Providing a reasonable representation of the notion of relevance in belief kinematics is the main challenge in this paper. For each of the two new Jeffrey's rules, we formalize a form of belief kinematics and characterize relevance in this belief kinematics by a *conditional belief function* on Ω with respect to the coarsening frame $\langle \Omega, \mathcal{B} \rangle$. Our definition of conditional belief functions differs from Dempster's rule of conditioning [SHAFFER, 1976] in that our definition depends on the coarsening frame $\langle \Omega, \mathcal{B} \rangle$ while Dempster's rule doesn't. Our conditional belief functions are natural generalizations of classical conditional probability functions and provide a measure of the relevance of any proposition in \mathcal{B} to all propositions in 2^Ω .

In this paper, we incorporate the above principle of belief kinematics into the two *new* Jeffrey's rules in the theory of belief functions by satisfying the following constraints:

- (Constraint 1) These two rules should be a natural generalization of Jeffrey's rule in probability theory, i.e., they should be the same as Jeffrey's rule in probability theory when the prior belief function *bel* is a probability function.
- (Constraint 2) On the coarsening frame $\langle \Omega, \mathcal{B} \rangle$, the posterior belief function *bel'* according to the rules should agree with the belief function *bel_e* that specifies the evidence.
- (Constraint 3) The revision rules should obey some natural evidence-theoretic generalization of probabil-

ity kinematics like the above belief kinematics.

Any Jeffrey's rule in DS theory should at least meet Constraint 1. Constraint 2 is Smets' distinguishing constraint **C1** [SMETS, 1993A]. Constraint 3 is the most important one and is the key point of this paper. We believe that this constraint for Jeffrey's rule in the theory of evidence is as important as the principle of probability kinematics is for Jeffrey's rule in probability theory. Unlike similar rules for belief functions found in the literature (See Section 5), our new Jeffrey's rules *naturally* transfer important properties of probabilistic belief revisions to the theory of evidence.

In order to show that the above revisions based on belief kinematics are optimal, we provide for each revision rule a distance measure for bounding belief changes due to the revisions and show that the belief function obtained according to the corresponding form of belief kinematics is the closest to the prior one among all belief functions satisfying Constraint 2.

2 JEFFREY'S RULE IN PROBABILITY THEORY

Let Pr be a probability function on a probability space $\langle \Omega, \mathcal{A} \rangle$ where \mathcal{A} is the Boolean algebra of subsets of Ω with the usual set operations. Suppose that new evidence suggests the desirability of revising Pr and that the total evidence determines a family \mathcal{E} of mutually exclusive and exhaustive subsets of Ω and a probability function Pr_e on the Boolean algebra \mathcal{B} of finite unions of elements of \mathcal{E} . Without loss of generality, we assume that $Pr_e(E) > 0$ for all $E \in \mathcal{E}$. The new posterior probability function Pr' on $\langle \Omega, \mathcal{A} \rangle$ proposed by *Jeffrey's rule* is as follows: for any $A \subseteq \Omega$,

$$Pr'(A) = \sum_{E \in \mathcal{E}} Pr(A|E)Pr_e(E) \quad (1)$$

A probability function Pr^* on the probability space $\langle \Omega, 2^\Omega \rangle$ is said to be obtained from Pr by the *principle of probability kinematics* on \mathcal{E} if, for any $E \in \mathcal{E}$,

$$Pr^*(A|E) = Pr(A|E) \text{ for every event } A \subseteq \Omega.$$

In other words, the principle of probability kinematics assumes that the conditional probability in every event A given any $E \in \mathcal{E}$ remains unchanged. This concept was proposed by Jeffrey [JEFFREY, 1983] to capture the notion that, even though Pr^* and Pr disagree on the probabilities of events in the *coarser* Boolean algebra \mathcal{B} , they agree on their *relevance* to every event A in \mathcal{A} .

Actually the above posterior probability function Pr' proposed by Jeffrey's rule in Eq. (1) is the *unique* probability revision Pr^* which satisfies the following two requirements [CHAN AND DARWICHE, 2003]:

- **(C1)**: probability kinematics on \mathcal{E} : for any $E \in \mathcal{B}$, $Pr^*(A|E) = Pr(A|E)$ for all $A \subseteq \Omega$;
- **(C2)**: $Pr^*(E) = Pr_e(E)$ for all $E \in \mathcal{B}$.

A distance measure D can be defined for probability functions as follows [CHAN AND DARWICHE, 2002]: for any two probability functions Pr_1 and Pr_2 ,

$$D(Pr_1, Pr_2) = \ln \max_{\omega \in \Omega} \frac{Pr_2(\omega)}{Pr_1(\omega)} - \ln \min_{\omega \in \Omega} \frac{Pr_2(\omega)}{Pr_1(\omega)}$$

Among all the probability functions that satisfy the above requirement **(C2)**, the posterior probability function Pr' proposed by Jeffrey's rule is the *closest* to Pr according to this distance measure [CHAN AND DARWICHE, 2003].

3 JEFFREY'S RULE IN DEMPSTER-SHAFER THEORY

3.1 BELIEF FUNCTIONS

Let Ω be a frame of discernment and $\mathcal{A} = 2^\Omega$ be the Boolean algebra of events. A *belief function* is a function $bel : \mathcal{A} \rightarrow [0, 1]$ satisfying the following conditions:

1. $bel(\emptyset) = 0$;
2. $bel(\Omega) = 1$; and
3. $bel(\bigcup_{i=1}^n A_i) = \sum_{\emptyset \neq I \subseteq \{1, \dots, n\}} (-1)^{|I|+1} bel(\bigcap_{i \in I} A_i)$ where $A_i \in \mathcal{A}$ for all $i \in \{1, \dots, n\}$.

A *mass assignment* (or *mass function*) is a mapping $m : \mathcal{A} \rightarrow [0, 1]$ satisfying

$$m(\emptyset) = 0, \sum_{A \in \mathcal{A}} m(A) = 1.$$

Shafer [SHAFFER, 1976] has shown that a mapping $f : \mathcal{A} \rightarrow [0, 1]$ is a belief function if and only if its Möbius transform is a mass assignment. In other words, if $m : \mathcal{A} \rightarrow [0, 1]$ is a mass assignment, then it determines a belief function $bel : \mathcal{A} \rightarrow [0, 1]$ as follows:

$$bel(A) = \sum_{B \subseteq A} m(B) \text{ for all } A \in \mathcal{A}.$$

Moreover, given a belief function bel , we can obtain its corresponding mass function m as follows:

$$m(A) = \sum_{B \subseteq A} (-1)^{|A \setminus B|} bel(B), \text{ for all } A \in \mathcal{A}.$$

Intuitively, for a subset event A , $m(A)$ measures the belief that an agent commits *exactly* to A , not the total belief that an agent commits to A .

3.2 JEFFREY'S RULE IN THE THEORY OF EVIDENCE

In order to introduce the principle of belief kinematics, we need to set up a setting in terms of *refinements and coarsenings* of frames of discernments. The idea that one frame Ω of discernment is obtained from another frame Θ of discernment by splitting some or all of the elements of Θ may be represented mathematically by specifying, for each $\theta \in \Theta$, the subset $\omega(\{\theta\})$ of Ω consisting of those possibilities into which θ has been split. For this representation to be sensible, we need only require that the sets $\omega(\{\theta\})$ should constitute a disjoint partition of Ω . Given such a disjoint partition $\omega(\{\theta\})$, we may set

$$\omega(A) = \bigcup_{\theta \in A} \omega(\{\theta\})$$

for each $A \subseteq \Theta$; $\omega(A)$ will consist of all the possibilities in Ω that are obtained by splitting the elements of A , and the mapping $\omega : 2^\Theta \rightarrow 2^\Omega$ that is thus defined will provide a thorough description of the splitting. Such a mapping ω is called a *refining*. Whenever $\omega : 2^\Theta \rightarrow 2^\Omega$ is a refining, we call Ω a *refinement* of Θ and Θ a *coarsening* of Ω .

In this paper, we are particularly interested in the case when Θ is the set of equivalence classes with respect to some partition Π of Ω . So the mapping $\omega(\{\Pi(w)\}) = \Pi(w)$ for each $w \in \Omega$ is a refining and Θ is a coarsening of Ω where $\Pi(w)$ is the equivalence class of w . We denote this special coarsening Θ of Ω as Ω/Π . On the other hand, Ω/Π may be regarded as a subalgebra \mathcal{B} of the powerset of Ω with the set of atoms of \mathcal{B} forming the partition Π of Ω . Our following definition of Jeffrey's rules in Dempster-Shafer theory is in terms of this type of presentation of the coarsening Ω/Π as $\langle \Omega, \mathcal{B} \rangle$. For example, $\Pi = \{\{w_1, w_2\}, \{w_3, w_4\}, \{w_5, w_6\}\}$ is a partition of $\Omega = \{w_1, w_2, w_3, w_4, w_5, w_6\}$. Then the associated subalgebra \mathcal{B} consists of the sets $\bigcup_{B \subseteq \Pi} B$ with the atoms $\{w_1, w_2\}$, $\{w_3, w_4\}$ and $\{w_5, w_6\}$ in \mathcal{B} .

For each $A \subseteq \Omega$, we define

$$\mathbf{B}(A) := \bigcap \{B \in \mathcal{B} : A \subseteq B\}$$

In other words, $\mathbf{B}(A)$ is the least element of \mathcal{B} that contains A as a subset and hence is called the *upper approximation* of A in \mathcal{B} [SMETS, 1993A]. For the above example, if $A = \{w_1, w_3, w_5\}$, then $\mathbf{B}(A) = \Omega$.

Let $\langle \Omega, \mathcal{B} \rangle$ be a coarsening of Ω where \mathcal{B} is a subalgebra of the powerset 2^Ω of Ω with its atoms forming a partition of Ω . Suppose that $bel : 2^\Omega \rightarrow [0, 1]$ is a belief function on Ω with m as its corresponding mass assignment. Then the *derived mass assignment* $m_{\mathcal{B}}^{in}$ on the coarsening $\langle \Omega, \mathcal{B} \rangle$ can be obtained through the following formula: for any $B \in \mathcal{B}$,

$$m_{\mathcal{B}}^{in}(B) = \sum_{\mathbf{B}(A)=B, A \subseteq \Omega} m(A)$$

It is easy to see that, in the coarsening frame $\langle \Omega, \mathcal{B} \rangle$, $m_{\mathcal{B}}^{in}(B)$ measures the belief that commits exactly to B , not to any subset of B in \mathcal{B} . Let $bel_{\mathcal{B}}^{in}$ denote the corresponding belief function. It is easy to check that, for any $B \in \mathcal{B}$, $bel_{\mathcal{B}}^{in}(B) = bel(B)$. Intuitively, $bel_{\mathcal{B}}^{in}$ is the derived belief function on the coarsening frame of discernment with less distinctions. The beliefs in the same propositions in these two different frames with different distinctions should be the same as each other [SMETS, 1993B]. Correspondingly, since the resolution degree of the attention of the coarsening frame decreases, the mass assignment m has to change into $m_{\mathcal{B}}^{in}$.

For any A and B such that $A \subseteq B \in \mathcal{B}$ and $A \subseteq \Omega$, let $m_{/B}(A)$ denote $\sum_{E \subseteq B} m(A \cup E)$ and

$$m_{\mathcal{B}}^{out}(B) = \sum_{\mathbf{B}(E)=B, E \subseteq \Omega} m_{/B}(E).$$

It is easy to see that, if B is an atom of the subalgebra \mathcal{B} , then $m_{\mathcal{B}}^{in}(B) = bel(B)$ and $m_{\mathcal{B}}^{out}(B) = pl(B)$. Now we define two different *conditional belief functions* $bel_{\mathcal{B}}^{in}(\cdot|B)$ and $bel_{\mathcal{B}}^{out}(\cdot|B)$ on a given $B \in \mathcal{B}$ according to the above two different definitions of mass functions $m_{\mathcal{B}}^{in}$ and $m_{\mathcal{B}}^{out}$ on \mathcal{B} , respectively: for any $A \subseteq \Omega$,

(1) (Inner conditioning)

$$bel_{\mathcal{B}}^{in}(A|B) := \begin{cases} \frac{\sum_{A' \subseteq A, \mathbf{B}(A')=B} m(A')}{m_{\mathcal{B}}^{in}(B)}, & \text{if } m_{\mathcal{B}}^{in}(B) \neq 0, \\ \frac{|\{A' \subseteq A : \mathbf{B}(A')=B\}|}{|\{A' \subseteq \Omega : \mathbf{B}(A')=B\}|}, & \text{if } m_{\mathcal{B}}^{in}(B) = 0 \end{cases}$$

(2) (Outer conditioning)

$$bel_{\mathcal{B}}^{out}(A|B) := \begin{cases} \frac{\sum_{A' \subseteq A, \mathbf{B}(A')=B} m_{/B}(A')}{m_{\mathcal{B}}^{out}(B)}, & \text{if } m_{\mathcal{B}}^{out}(B) \neq 0, \\ \frac{|\{A' \subseteq A : \mathbf{B}(A' \cap B)=B\}|}{|\{A' \subseteq \Omega : \mathbf{B}(A' \cap B)=B\}|}, & \text{if } m_{\mathcal{B}}^{out}(B) = 0 \end{cases}$$

Note that both $bel_{\mathcal{B}}^{in}(\cdot|B)$ and $bel_{\mathcal{B}}^{out}(\cdot|B)$ are belief functions on 2^Ω for any $B \in \mathcal{B}$.

The superscripts *in* and *out* in the above notations are designated for the following two proposed revision rules: inner revision and outer revision. In particular, when B is an atom in the algebra \mathcal{B} , the above defined $bel_{\mathcal{B}}^{in}(\cdot|B)$ and $bel_{\mathcal{B}}^{out}(\cdot|B)$, are essentially the geometric conditionalization and the Dempster conditionalization of A on B , respectively. However, the essential difference of our above definitions of conditional belief functions from the well-known Dempster's rule of conditioning $bel(\cdot|B)$ [SHAFFER, 1976] is that they depend on the coarsening frame and hence on the degree of resolution of the attention while Dempster's rule of conditioning does not and is derived from Dempster's rule of combination. Moreover, Dempster's rule of combination relies on a basic assumption that the combined evidences (or beliefs) play the same

role and hence the combination operation is *symmetric*. In contrast, in our study of revision of beliefs using uncertain evidence, we treat uncertain evidence in terms of *effect* it has on beliefs once accepted, which is a function of both evidence strength and beliefs held before the evidence is obtained. Hence the prior beliefs and uncertain evidence are intrinsically asymmetric. In this sense, our definition is a natural generalization of the classical Bayesian definition of conditional probabilities.

Lemma 3.1 *Let bel be a belief function on Ω with m its corresponding mass assignment and $\langle \Omega, \mathcal{B} \rangle$ a coarsening as above.*

1. *For any $A \subseteq \Omega$,*

$$bel(A) = \sum_{B \in \mathcal{B}} bel_{\mathcal{B}}^{in}(A|B)m_B^{in}(B);$$

2. *If m_e is a mass assignment on $\langle \Omega, \mathcal{B} \rangle$, then the function $bel' : 2^\Omega \rightarrow [0, 1]$ defined as follows, for any $A \subseteq \Omega$,*

$$bel'(A) = \sum_{B \in \mathcal{B}} bel_{\mathcal{B}}^{in}(A|B)m_e(B)$$

is a belief function. In particular, for each $B \in \mathcal{B}$,

$$bel'(B) = \sum_{B' \in \mathcal{B}, B' \subseteq B} m_e(B').$$

In other words, if bel_e is the corresponding belief function of m_e on $\langle \Omega, \mathcal{B} \rangle$, then $bel'(B) = bel_e(B)$ for all $B \in \mathcal{B}$; namely, m_e is exactly the derived mass assignment $(m')_{\mathcal{B}}^{in}$ of bel' on the coarsening frame $\langle \Omega, \mathcal{B} \rangle$.

Proof. The first part is obvious. And the second part follows from the following fact: for any $B, B' \in \mathcal{B}$,

$$bel_{\mathcal{B}}^{in}(B'|B) := \begin{cases} 1, & \text{if } B \subseteq B' \\ 0 & \text{otherwise.} \end{cases}$$

QED

Lemma 3.2 *Let bel be a belief function on Ω with m its corresponding mass assignment and $\langle \Omega, \mathcal{B} \rangle$ a coarsening as above. If m_e is a mass assignment on $\langle \Omega, \mathcal{B} \rangle$, then the function $bel' : 2^\Omega \rightarrow [0, 1]$ defined as follows, for any $A \subseteq \Omega$,*

$$bel'(A) = \sum_{B \in \mathcal{B}} bel_{\mathcal{B}}^{out}(A|B)m_e(B)$$

is a belief function. In particular, for each $B \in \mathcal{B}$,

$$bel'(B) = \sum_{B' \in \mathcal{B}, B' \subseteq B} m_e(B').$$

In other words, if bel_e is the corresponding belief function of m_e on $\langle \Omega, \mathcal{B} \rangle$, then $bel'(B) = bel_e(B)$ for all $B \in \mathcal{B}$; namely, m_e is exactly the derived mass assignment $(m')_{\mathcal{B}}^{in}$ of bel' on the coarsening frame $\langle \Omega, \mathcal{B} \rangle$.

Proof. The first part is clear and the second follows from the following fact: for any $B, B' \in \mathcal{B}$,

$$bel_{\mathcal{B}}^{out}(B'|B) := \begin{cases} 1, & \text{if } B \subseteq B' \\ 0 & \text{otherwise.} \end{cases}$$

QED

Consider the problem of revising the belief function bel given uncertain evidence relating to a coarsening of Ω , which is represented as $\langle \Omega, \mathcal{B} \rangle$. One method of specifying the uncertain evidence is through the *effect* that it would have on beliefs once accepted. Specifically, according to the method, we have to specify uncertain evidence by providing the following constraint:

$$m'(B) = q_B, \text{ for each } B \in \mathcal{B} \quad (2)$$

where m' denotes the corresponding mass assignment of the new belief function bel' that results from accepting the given evidence. Also the specification can be represented as another belief function bel_e on $\langle \Omega, \mathcal{B} \rangle$ with m_e its corresponding mass assignment such that $m_e(B) = q_B$ for all $B \in \mathcal{B}$. To revise the belief function bel , we must therefore choose a *unique posterior* belief function that satisfies the above constraint. In order to achieve the uniqueness, we define next two forms of *belief kinematics*, the evidence-theoretic counterpart of the well-known probability kinematics [JEFFREY, 1983].

Definition 3.3 Suppose that bel and bel' are two belief functions on Ω , and m and m' are their corresponding mass assignments. Let $\langle \Omega, \mathcal{B} \rangle$ be a coarsening of Ω . The belief function bel' is said to be obtained from bel by *inner belief kinematics* on $\langle \Omega, \mathcal{B} \rangle$ if, for any $B \in \mathcal{B}$,

$$(bel')_{\mathcal{B}}^{in}(A|B) = bel_{\mathcal{B}}^{in}(A|B) \text{ for all } A \subseteq \Omega; \quad (3)$$

and it is said to be obtained from bel by *outer belief kinematics* on $\langle \Omega, \mathcal{B} \rangle$ if, for any $B \in \mathcal{B}$,

$$(bel')_{\mathcal{B}}^{in}(A|B) = bel_{\mathcal{B}}^{out}(A|B) \text{ for all } A \subseteq \Omega. \quad (4)$$

◁

Intuitively, the above principle of belief kinematics on $\langle \Omega, \mathcal{B} \rangle$ says that, even though bel and bel' may disagree on propositions on $\langle \Omega, \mathcal{B} \rangle$, they agree on their relevance to every event $A \subseteq \Omega$.

Now we define two revisions proposed by *Jeffrey's rule* as follows: for any $A \subseteq \Omega$,

$$1. \ bel'(A) = \sum_{B \in \mathcal{B}} bel_{\mathcal{B}}^{in}(A|B)q_B; \text{ and}$$

$$2. \ bel'(A) = \sum_{B \in \mathcal{B}} bel_{\mathcal{B}}^{out}(A|B)q_B$$

According to Lemmas 3.1 and 3.2, both revisions satisfy the above constraint (2), and are indeed belief functions. These two revisions are called *inner and outer revisions* and the resulting belief functions are denoted as $bel^{in'}$ and $bel^{out'}$ by adding the corresponding superscripts in and out , respectively. It is easy to see that the well-known Jeffrey's rule for probability functions is a special case of our more general rules here for belief functions. So, our Jeffrey's rules satisfy Constraint 1.

Theorem 3.4 *The new belief function $bel^{in'}$ given above is the one and only belief function that satisfies the constraint in Eq. (2) and that is obtained from bel by inner belief kinematics on the coarsening frame $\langle \Omega, \mathcal{B} \rangle$.*

Proof. According to Lemma 3.1, it suffices to show that the new posterior belief function $bel^{in'}$ obtained through Jeffrey's rule satisfies the condition for inner belief kinematics: for any $A \subseteq \Omega, B \in \mathcal{B}$,

$$bel_B^{in}(A|B) = (bel^{in'})_B^{in}(A|B)$$

First note that, for any $A \subseteq \Omega$,

$$m^{in'}(A) = \sum_{B \in \mathcal{B}} \frac{q_B}{m_B^{in}(B)} l_B^{in}(A)$$

where $m^{in'}$ is the corresponding mass function of $bel^{in'}$ and

$$l_B^{in}(A) := \begin{cases} m(A), & \text{if } \mathbf{B}(A) = B \\ 0 & \text{otherwise.} \end{cases}$$

This follows directly from the following reasoning:

$$\begin{aligned} \sum_{A' \subseteq A} m^{in'}(A') &= \sum_{A' \subseteq A} \left(\sum_{B \in \mathcal{B}} \frac{q_B}{m_B^{in}(B)} l_B^{in}(A') \right) \\ &= \sum_{B \in \mathcal{B}} \left(\sum_{A' \subseteq A} \frac{q_B}{m_B^{in}(B)} l_B^{in}(A') \right) \\ &= \sum_{B \in \mathcal{B}} \left(\frac{\sum_{A' \subseteq A} l_B^{in}(A')}{m_B^{in}(B)} q_B \right) \\ &= \sum_{B \in \mathcal{B}} \left(\frac{\sum_{A' \subseteq A, \mathbf{B}(A')=B} m(A')}{m_B^{in}(B)} q_B \right) \\ &= \sum_{B \in \mathcal{B}} bel_B^{in}(A|B) q_B \\ &= bel^{in'}(A) \end{aligned}$$

According to Lemma 3.1, $(m^{in'})_B^{in}(B) = \sum_{\mathbf{B}(A)=B, A \subseteq \Omega} m^{in'}(A) = q_B$ where $(m^{in'})_B^{in}$ is the derived mass assignment of $m^{in'}$ on $\langle \Omega, \mathcal{B} \rangle$. Next we use

this expression of $m^{in'}$ to proceed as follows:

$$\begin{aligned} (bel^{in'})_B^{in}(A|B) &= \frac{1}{q_B} \sum_{A' \subseteq A, \mathbf{B}(A')=B} m^{in'}(A') \\ &= \frac{1}{q_B} \sum_{A' \subseteq A, \mathbf{B}(A')=B} \sum_{B' \in \mathcal{B}} \frac{q_{B'} l_{B'}^{in}(A')}{m_B^{in}(B')} \\ &= \frac{1}{q_B} \sum_{A' \subseteq A, \mathbf{B}(A')=B} \frac{q_B}{m_B^{in}(B)} m(A') \\ &= \frac{\sum_{A' \subseteq A, \mathbf{B}(A')=B} m(A')}{m_B^{in}(B)} \\ &= bel_B^{in}(A|B) \end{aligned}$$

QED

Theorem 3.5 *The new belief function $bel^{out'}$ given above is the one and only belief function that satisfies the constraint in Eq. (2) and that is obtained from bel by outer belief kinematics on the coarsening frame $\langle \Omega, \mathcal{B} \rangle$.*

Proof. According to Part (1) of Lemma 3.1 and Lemma 3.2, it suffices to show that the new posterior belief function obtained through Jeffrey's rule satisfies the condition in outer belief kinematics, i.e., for any $B \in \mathcal{B}$,

$$(bel^{out'})_B^{in}(A|B) = bel_B^{out}(A|B) \text{ for all } A \subseteq \Omega.$$

But this follows from a similar argument to that in the proof of Theorem 3.4.

QED

The above propositions tell us that the two Jeffrey's rules are obtained from belief kinematics and hence satisfies Constraint 3.

Example 3.6 The following example is adapted from the original one by Jeffrey [JEFFREY, 1983] (also [CHAN AND DARWICHE, 2003]). Assume that we are given a piece of cloth, where its color can be one of: green, blue, or violet. We want to know whether, on the next day, the cloth will be sold, or not sold. We denote the possible states as follows:

$$\begin{aligned} w_{1,g} &= (\text{sold}, \text{green}), & w_{0,g} &= (\text{not sold}, \text{green}) \\ w_{1,b} &= (\text{sold}, \text{blue}), & w_{0,b} &= (\text{not sold}, \text{blue}) \\ w_{1,v} &= (\text{sold}, \text{violet}), & w_{0,v} &= (\text{not sold}, \text{violet}) \end{aligned}$$

Our original belief bel is given by the following mass assignment m on $\Omega := \{w_{n,c} : n \in \{0, 1\}, c \in \{b, g, v\}\}$:

$$\begin{aligned} m(\{w_{1,g}\}) &= m(\{w_{1,b}\}) = m(\{w_{1,v}\}) = 0.1 \\ m(\{w_{0,g}\}) &= m(\{w_{0,b}\}) = m(\{w_{0,v}\}) = 0.15 \\ m(\{w_{1,g}, w_{0,b}\}) &= m(\{w_{1,b}, w_{0,v}\}) = 0.1 \\ m(\{w_{1,g}, w_{1,v}\}) &= 0.05 \end{aligned}$$

The possible states w_c of colors denote $\{w_{1,c}, w_{0,c}\}$ for all $c \in \{g, v, b\}$. Let \mathcal{B} be a subalgebra of the powerset of Ω that consists of the propositions of the form $\bigcup_{B \subseteq \{w_b, w_g, w_v\}} B$. It is easy to see that $\langle \Omega, \mathcal{B} \rangle$ is a coarsening of $\langle \Omega, 2^\Omega \rangle$.

The derived mass assignment $m_{\mathcal{B}}^{in}$ on the coarsening frame $\langle \Omega, \mathcal{B} \rangle$ can be computed as follows:

$$\begin{aligned} m_{\mathcal{B}}^{in}(\{w_g\}) &= m_{\mathcal{B}}^{in}(\{w_b\}) = m_{\mathcal{B}}^{in}(\{w_v\}) = 0.25 \\ m_{\mathcal{B}}^{in}(\{w_g, w_b\}) &= m_{\mathcal{B}}^{in}(\{w_b, w_v\}) = 0.1 \\ m_{\mathcal{B}}^{in}(\{w_g, w_v\}) &= 0.05, \quad m_{\mathcal{B}}^{in}(\{w_b, w_g, w_v\}) = 0 \end{aligned}$$

Now we consider the conditional beliefs of a given proposition $A := \{w_{1,g}, w_{0,b}, w_{1,v}\}$ as an illustration. We obtain $bel(A) = 0.5$. According to our previous definition of inner conditioning, we have

$$\begin{aligned} bel_{\mathcal{B}}^{in}(A|\{w_g\}) &= \frac{2}{5}, & bel_{\mathcal{B}}^{in}(A|\{w_b\}) &= \frac{3}{5} \\ bel_{\mathcal{B}}^{in}(A|\{w_v\}) &= \frac{2}{5}, & bel_{\mathcal{B}}^{in}(A|\{w_g, w_b\}) &= 1 \\ bel_{\mathcal{B}}^{in}(A|\{w_g, w_v\}) &= 1, & bel_{\mathcal{B}}^{in}(A|\{w_b, w_v\}) &= 0 \\ bel_{\mathcal{B}}^{in}(A|\{w_b, w_v, w_g\}) &= \frac{1}{27} \end{aligned}$$

Assume that we now inspect the cloth by candlelight, and conclude that our belief on the color of the cloth should be:

$$\begin{aligned} bel_e(\{w_g\}) &= bel_e(\{w_b\}) = bel_e(\{w_v\}) = 0.2 \\ bel_e(\{w_g, w_b\}) &= bel_e(\{w_b, w_v\}) = 0.5 \\ bel_e(\{w_g, w_v\}) &= 0.6 \end{aligned}$$

The corresponding mass assignment m_e is as follows:

$$\begin{aligned} m_e(\{w_g\}) &= m_e(\{w_b\}) = m_e(\{w_v\}) = 0.2 \\ m_e(\{w_g, w_b\}) &= m_e(\{w_b, w_v\}) = 0.1 \\ m_e(\{w_g, w_v\}) &= 0.2 \end{aligned}$$

So, according to our definition of Jeffrey's rule, we have the new inner revision of belief in the event A :

$$bel^{in'}(A) = \sum_{B \in \mathcal{B}} bel_{\mathcal{B}}^{in}(A|B)m_e(B) = 0.58$$

Now we compute the outer revision of the belief in A . The mass assignment $m_{\mathcal{B}}^{out}$ on the coarsening frame $\langle \Omega, \mathcal{B} \rangle$ can be computed as follows:

$$\begin{aligned} m_{\mathcal{B}}^{out}(\{w_g\}) &= 0.4 = m_{\mathcal{B}}^{out}(\{w_v\}), m_{\mathcal{B}}^{out}(\{w_b\}) = 0.45 \\ m_{\mathcal{B}}^{out}(\{w_g, w_b\}) &= 0.1 = m_{\mathcal{B}}^{out}(\{w_b, w_v\}) \\ m_{\mathcal{B}}^{out}(\{w_g, w_v\}) &= 0.05, \quad m_{\mathcal{B}}^{out}(\{w_b, w_g, w_v\}) = 0 \end{aligned}$$

According to our previous definition of outer conditioning,

we have

$$\begin{aligned} bel_{\mathcal{B}}^{out}(A|\{w_g\}) &= \frac{1}{4}, bel_{\mathcal{B}}^{out}(A|\{w_b\}) = \frac{2}{9} \\ bel_{\mathcal{B}}^{out}(A|\{w_v\}) &= \frac{1}{8}, bel_{\mathcal{B}}^{out}(A|\{w_g, w_b\}) = 1 \\ bel_{\mathcal{B}}^{out}(A|\{w_g, w_v\}) &= 1, bel_{\mathcal{B}}^{out}(A|\{w_b, w_v\}) = 0 \\ bel_{\mathcal{B}}^{out}(A|\{w_b, w_v, w_g\}) &= \frac{1}{27} \end{aligned}$$

Hence we have

$$bel^{out'}(A) = \sum_{B \in \mathcal{B}} bel_{\mathcal{B}}^{out}(A|B)m_e(B) = \frac{151}{360}$$

4 MEASURES FOR BOUNDING BELIEF CHANGES

One important question relating to belief revision is that of measuring the extent to which a revision disturbs existing beliefs. In the following, we simulate the work by Chan and Darwiche [CHAN AND DARWICHE, 2002] by proposing for each Jeffrey's revision rule a distance measure for belief functions which can be used to bound the amount of belief changes induced by this revision using uncertain evidence and show that, according to this measure, the posterior belief function obtained by the corresponding belief kinematics is the closest to the original one among all belief functions that satisfy the constraint in Eq. (2).

Definition 4.1 Let bel and bel' be two belief functions over the same frame Ω of discernment. We define a measure between bel and bel' as follows:

$$D^{in}(bel, bel') = \ln \max_{A \subseteq \Omega} \frac{m'(A)}{m(A)} - \ln \min_{A \subseteq \Omega} \frac{m'(A)}{m(A)}$$

where $\frac{0}{0}$ is defined to be 1. It is easy to check that D^{in} is a distance (or metric), satisfying the three properties of distance and, whenever there is a subset A for which $m(A) = 0$ and $m'(A) > 0$ or vice versa, the distance $D^{in}(bel, bel')$ for the corresponding belief functions is equal to infinity. \triangleleft

Lemma 4.2 Let $\langle \Omega, \mathcal{B} \rangle$ be a coarsening of $\langle \Omega, 2^\Omega \rangle$. Assume that $bel^{in'}$ is obtained from bel by applying Jeffrey's rule according to inner belief kinematics (Eq. (3)), given the uncertain evidence specified by the set of posterior beliefs $(m^{in'})_{\mathcal{B}}^{in}(B) = q_B$, for $B \in \mathcal{B}$ where $(m^{in'})_{\mathcal{B}}^{in}$ is the derived mass assignment of $m^{in'}$, the corresponding mass assignment of $bel^{in'}$, on $\langle \Omega, \mathcal{B} \rangle$.

1. For any $A \subseteq \Omega$, if $\mathbf{B}(A) = B$, then

$$\frac{m^{in'}(A)}{m(A)} = \frac{(m^{in'})_{\mathcal{B}}^{in}(B)}{m_{\mathcal{B}}^{in}(B)}.$$

2. The distance $D^{in}(bel, bel')$ between bel and $bel^{in'}$ is given by

$$D^{in}(bel, bel^{in'}) = \ln \max_{B \in \mathcal{B}} \frac{q_B}{m_B^{in}(B)} - \ln \min_{B \in \mathcal{B}} \frac{q_B}{m_B^{in}(B)}$$

Proof. The first part follows from the following observation: for any $A \subseteq \Omega$ and $B \in \mathcal{B}$,

$$\begin{aligned} \frac{\sum_{\mathbf{B}(A')=B, A' \subseteq A} m^{in'}(A')}{\sum_{\mathbf{B}(A')=B, A' \subseteq A} m(A')} &= \frac{(bel^{in'})_B^{in}(A|B)(m^{in'})_B^{in}(B)}{bel_B^{in}(A|B)m_B^{in}(B)} \\ &= \frac{(m^{in'})_B^{in}(B)}{m_B^{in}(B)} \end{aligned}$$

The second equality comes from the condition for inner belief kinematics. QED

The following theorem says that the principle of inner belief kinematics can be viewed as a principle for minimizing belief change with respect to the metric D^{in} .

Theorem 4.3 *For the belief functions bel and $bel^{in'}$ in Lemma 4.2, $bel^{in'}$ is the closest to bel according to the above distance measure D^{in} among all possible belief functions that agree with $bel^{in'}$ on the propositions in the subalgebra \mathcal{B} .*

Proof. Suppose that m is the corresponding mass assignment of bel . Let bel'' be any belief function with m'' as its corresponding mass assignment that satisfies the constraint: $(m'')_B^{in}(B) = (m^{in'})_B^{in}(B)$ for all $B \in \mathcal{B}$. Let $B_{max} = \arg\max_{B \in \mathcal{B}} (\frac{(m^{in'})_B^{in}(B)}{m_B^{in}(B)})$ and $B_{min} = \arg\min_{B \in \mathcal{B}} (\frac{(m^{in'})_B^{in}(B)}{m_B^{in}(B)})$. Define $r_{max} = \max_{A \subseteq \Omega} \frac{m''(A)}{m(A)}$. Then we have the following inequality:

$$\begin{aligned} r_{max} m_B^{in}(B_{max}) &= r_{max} \sum_{\mathbf{B}(A)=B_{max}, A \subseteq \Omega} m(A) \\ &\geq \sum_{\mathbf{B}(A)=B_{max}, A \subseteq \Omega} \frac{m''(A)}{m(A)} m(A) \\ &= \sum_{\mathbf{B}(A)=B_{max}, A \subseteq \Omega} m''(A) \\ &= (m'')_B^{in}(B_{max}) \\ &= (m^{in'})_B^{in}(B_{max}) \end{aligned}$$

So we have shown that $r_{max} \geq \frac{(m^{in'})_B^{in}(B_{max})}{m_B^{in}(B_{max})}$. Similarly, we can define $r_{min} = \min_{A \subseteq \Omega} \frac{m''(A)}{m(A)}$ and show that $r_{min} \leq \frac{(m^{in'})_B^{in}(B_{min})}{m_B^{in}(B_{min})}$. Therefore, the distance measure

between bel and bel'' is:

$$\begin{aligned} D^{in}(bel, bel'') &= \ln r_{max} - \ln r_{min} \\ &\geq \ln \frac{(m^{in'})_B^{in}(B_{max})}{m_B^{in}(B_{max})} - \ln \frac{(m^{in'})_B^{in}(B_{min})}{m_B^{in}(B_{min})} \\ &= \ln \max_{B \in \mathcal{B}} \frac{(m^{in'})_B^{in}(B)}{m_B^{in}(B)} - \ln \min_{B \in \mathcal{B}} \frac{(m^{in'})_B^{in}(B)}{m_B^{in}(B)} \\ &= D^{in}(bel, bel') \end{aligned}$$

The last equality follows from Lemma 4.2.

QED

Now we define a distance D^{out} for the outer revision. The essential difference of D^{out} from the above D^{in} for the inner revision is that D^{out} depends on the associated coarsening frame.

Definition 4.4 Let bel and bel' be two belief functions over the same frame Ω of discernment. We define a measure between bel and bel' with respect to a coarsening $\langle \Omega, \mathcal{B} \rangle$ as follows:

$$D^{out}(bel, bel') = \ln \max_{A \subseteq \Omega} \frac{m'(A)}{m_{/\mathbf{B}(A)}(A)} - \ln \min_{A \subseteq \Omega} \frac{m'(A)}{m_{/\mathbf{B}(A)}(A)}$$

where $\frac{0}{0}$ is defined to be 1. It is easy to check that D^{out} is a distance (or metric), satisfying the three properties of distance. \triangleleft

Lemma 4.5 Let $\langle \Omega, \mathcal{B} \rangle$ be a coarsening of $\langle \Omega, 2^\Omega \rangle$. Assume that $bel^{out'}$ is obtained from bel by applying Jeffrey's rule according to the outer belief kinematics, given the uncertain evidence specified by the set of posterior beliefs $(m^{out'})_B^{in}(B) = q_B$, for $B \in \mathcal{B}$ where $(m^{out'})_B^{in}$ is the derived mass assignment of $m^{out'}$, the corresponding mass assignment of $bel^{out'}$, on $\langle \Omega, \mathcal{B} \rangle$.

1. For any $A \subseteq \Omega$, if $\mathbf{B}(A) = B$, then

$$\frac{m^{out'}(A)}{m_{/\mathbf{B}(A)}(A)} = \frac{(m^{out'})_B^{in}(B)}{m_B^{out}(B)}.$$

2. The distance $D^{out}(bel, bel')$ between bel and bel' is given by

$$D^{out}(bel, bel') = \ln \max_{B \in \mathcal{B}} \frac{q_B}{m_B^{out}(B)} - \ln \min_{B \in \mathcal{B}} \frac{q_B}{m_B^{out}(B)}$$

Proof. The first part follows from the following observation: for any $A \subseteq \Omega$ and $B \in \mathcal{B}$,

$$\begin{aligned} \frac{\sum_{\mathbf{B}(A')=B, A' \subseteq A} m^{out'}(A')}{\sum_{\mathbf{B}(A')=B, A' \subseteq A} m_{/\mathbf{B}(A')}(A')} &= \frac{(bel^{out'})_B^{in}(A|B)(m^{out'})_B^{in}(B)}{bel_B^{out}(A|B)m_B^{out}(B)} = \frac{(m^{out'})_B^{in}(B)}{m_B^{out}(B)} \end{aligned}$$

The second equality comes from the condition for outer belief kinematics. QED

The following theorem says that the principle of outer belief kinematics can be viewed as a principle for minimizing belief change with respect to D^{out} .

Theorem 4.6 *For the belief functions bel and $bel^{out'}$ in Lemma 4.5, $bel^{out'}$ is the closest to bel according to the above distance measure D^{out} among all possible belief functions that agree with $bel^{out'}$ on the propositions in the subalgebra \mathcal{B} .*

Proof. Suppose that m is the corresponding mass assignments of bel . Let bel'' be any belief function with m'' as its corresponding mass assignment that satisfies the constraint: $(m'')_{\mathcal{B}}^{in}(B) = (m^{out'})_{\mathcal{B}}^{in}(B)$ for all $B \in \mathcal{B}$. Let $B_{max} = \operatorname{argmax}_{B \in \mathcal{B}} (\frac{(m^{out'})_{\mathcal{B}}^{in}(B)}{m_{\mathcal{B}}^{out}(B)})$ and $B_{min} = \operatorname{argmin}_{B \in \mathcal{B}} (\frac{(m^{out'})_{\mathcal{B}}^{in}(B)}{m_{\mathcal{B}}^{out}(B)})$. Define $r_{max} = \max_{A \subseteq \Omega} \frac{m''(A)}{m_{/\mathcal{B}(A)}(A)}$. Then we have the following inequality:

$$\begin{aligned} r_{max} m_{\mathcal{B}}^{out}(B_{max}) &= r_{max} \sum_{\mathcal{B}(A)=B_{max}, A \subseteq \Omega} m_{/\mathcal{B}(A)}(A) \\ &\geq \sum_{\mathcal{B}(A)=B_{max}, A \subseteq \Omega} \frac{m''(A)}{m_{/\mathcal{B}(A)}(A)} m_{/\mathcal{B}(A)}(A) \\ &= \sum_{\mathcal{B}(A)=B_{max}, A \subseteq \Omega} m''(A) \\ &= (m'')_{\mathcal{B}}^{in}(B_{max}) \\ &= (m^{out'})_{\mathcal{B}}^{in}(B_{max}) \end{aligned}$$

So we have shown that $r_{max} \geq \frac{(m^{out'})_{\mathcal{B}}^{in}(B_{max})}{m_{\mathcal{B}}^{out}(B_{max})}$. Similarly, we can define $r_{min} = \min_{A \subseteq \Omega} \frac{m''(A)}{m_{/\mathcal{B}(A)}(A)}$ and show that $r_{min} \leq \frac{(m^{out'})_{\mathcal{B}}^{in}(B_{min})}{m_{\mathcal{B}}^{out}(B_{min})}$. Therefore, the distance measure between bel and bel'' is:

$$\begin{aligned} D^{out}(bel, bel'') &= \ln \max_{A \subseteq \Omega} \frac{m''(A)}{m_{/\mathcal{B}(A)}(A)} - \ln \min_{A \subseteq \Omega} \frac{m''(A)}{m_{/\mathcal{B}(A)}(A)} \\ &= \ln r_{max} - \ln r_{min} \\ &\geq \ln \frac{(m^{out'})_{\mathcal{B}}^{in}(B_{max})}{m_{\mathcal{B}}^{out}(B_{max})} - \ln \frac{(m^{out'})_{\mathcal{B}}^{in}(B_{min})}{m_{\mathcal{B}}^{out}(B_{min})} \\ &= \ln \max_{B \in \mathcal{B}} \frac{(m^{out'})_{\mathcal{B}}^{in}(B)}{m_{\mathcal{B}}^{out}(B)} - \ln \min_{B \in \mathcal{B}} \frac{(m^{out'})_{\mathcal{B}}^{in}(B)}{m_{\mathcal{B}}^{out}(B)} \\ &= D^{out}(bel, bel') \end{aligned}$$

The last equality follows from Lemma 4.5.

QED

Example 4.7 Now, by using Lemmas 4.2 and 4.5, we compute the distances between bel and bel' in Example 3.6:

$$\begin{aligned} D^{in}(bel, bel') &= \ln 1 - \ln \frac{1}{4} \\ &= 2 \ln 2 \\ D^{out}(bel, bel') &= \ln 4 - \ln \frac{4}{9} \\ &= 3 \ln 3 \end{aligned}$$

5 RELATED WORKS AND CONCLUSION

Although belief revision in probability theory is fully studied and researchers have generally agreed on the standard form of Jeffrey's rule, the corresponding revision rule in evidence theory has seldom been adequately addressed and there is not yet any standard form of this rule that has been universally recognized. Although all the forms of Jeffrey's rule in the theory of evidence in the literature are generalizations of this rule in probability theory, none of them satisfies all of the three natural constraints proposed in this paper. Usually they satisfy some constraints but do not satisfy the others. In particular, none of these forms in the literature has considered the revision of beliefs using uncertain evidence from the perspective in this paper viewing Jeffrey's rule as a form of the evidence-theoretic counterpart of probability kinematics, which should be the essence of Jeffrey's rule in Dempster-Shafer theory [WAGNER, 1992]. Moreover, none of those Jeffrey's rules in DS -theory in the literature has provided any distance measures for bounding belief changes due to revision and our work is the first to achieve that. Our distance measures for belief functions are adapted from the one for revision of probabilistic beliefs using uncertain evidence as *virtual certain evidence* according to Pearl's method [PEARL, 1988] and hence different from those distances for belief functions in the literature [JOUSSELME AND MAUPIN, 2012].

Jeffrey's rules for belief functions in the literature are proposed from different perspectives. Shafer [SHAFER, 1981] has studied Jeffrey's rule. He proposed that its generalization can be found in Dempster's rule of combination. His proposal doesn't fit with Constraint 2 and we agree with Smets [SMETS, 1993A] that Constraint 2 is more important in the spirit of Jeffrey's updating than Shafer's proposal. In addition, Wagner [WAGNER, 1992] studied Jeffrey's rule in evidence theory from the perspective of viewing belief functions as lower envelopes. So his perspective is quite different from our proposal in term of mass functions.

Dubois and Prade [DUBOIS AND PRADE, 1991, DUBOIS AND PRADE, 1993] investigated updating and revision rules in a variety of uncertainty models including belief functions. They proposed the following form of Jeffrey's rule in DS theory:

$bel'(A) = \sum_{B \in \mathcal{B}} \frac{bel(A \cup \bar{B}) - bel(\bar{B})}{pl(\bar{B})} m_e(B)$. This form is one of several Jeffrey's rules studied by Ichihashi and Tanaka ([ICHIHASHI AND TANAKA, 1989]). Ma and others ([MA ET AL., 2010] and in more detail [MA ET AL., 2011]) proposed three different revision rules, namely the inner, outer and modified outer revisions. In particular, their modified outer revision generalizes Jeffrey's rule of updating in probability theory, Dempster's rule of conditioning and a form of AGM revision. Their rules work in a more general setting when the incoming input is a *general* mass function. They consider the *information content* associated with an epistemic state represented by some belief function rather than the *full specification* as in our paper. A belief function bel_1 is less informed than another one bel_2 if bel_2 is a specialization of bel_1 . They formalize the success postulate as requiring that the posterior belief function bel' be a specialization of the prior one bel . According to their viewpoint, if bel_1 and bel_2 are both defined on the same algebra \mathcal{A} and bel_1 is a specialization of bel_2 , then they are considered to be consistent with each other. However, according to our idea, they are inconsistent if they are not the same. We take the readaptation as revision [SMETS, 1993A]. Halpern [HALPERN, 2005] provided another form of belief-function revision rule: $bel'(A) = \sum_{i=1}^n bel_e(B_i) bel(A|B_i)$ where $(B_i)_{i=1}^n$ is a family of mutually exclusive and exhaustive subsets of Ω . His generalization is in terms of belief functions instead of mass functions. So it is quite different from ours. Moreover, bel' in his revision rule is not necessarily a belief function unless bel_e is a probability function. But, according to our proposal, uncertain evidence should be specified by a belief function bel_e . None of the above mentioned forms of Jeffrey's rules in *DS*-theory satisfies Constraint 2.

Our proposed Jeffrey's rules actually *improve* the two rules called source-conditioning and data-conditioning by Smets [SMETS, 1993A] especially the source-conditioning rule there. The motivation for the rule is not well justified and Smets' constraints for this rule are not well-defined. In Smets' Constraint **C2F**, bel should satisfy the requirement that, for any $X, Y \subseteq \Omega$, if $\mathbf{B}(X) = \mathbf{B}(Y)$, $\frac{bel(X)}{bel(Y)} = \frac{bel'(X)}{bel'(Y)}$. But generally bel' does not satisfy this requirement. Consider the above Example 3.6 and the proposition A . Let $A' = \{w_{0,g}, w_{1,b}, w_{0,v}\}$. Obviously, $\mathbf{B}(A) = \mathbf{B}(A') = \Omega$. However, $\frac{bel(A)}{bel(A')} = 1 \neq \frac{29}{18} = \frac{bel'(A)}{bel'(A')}$. The two forms of belief kinematics in this paper correct and improve the two constraints **C2F** and **C3F** in [SMETS, 1993A] (Parts (1) of Lemmas 4.2 4.5), respectively. Benferhat and others [BENFERHAT ET AL., 2011] studied Jeffrey's rule in a *possibilistic* framework using the possibilistic counterparts of probability kinematics, which is similar to our approach in this paper. But, our theory for belief functions here covers their approach in the quantitative possibilistic setting.

The following are some other constraints for defining Jeffrey's rules in Dempster-Shafer theory [MA ET AL., 2011]:

- (Constraint 4) When the incoming information is certain, the proposed Jeffrey's rule should be the same as Dempster's rule of conditioning.
- (Constraint 5) The proposed rule should satisfy some natural form of minimal change principle.
- (Constraint 6) The revision rule should preserve the new evidence.

We summarize our contributions in this paper by listing in a table the above major proposed Jeffrey's rules and their satisfied constraints:

Table 1: Summary

Constraint \ Rule	1	2	3	4	5	6
Shafer's rule	✓			✓		✓
Modified outer revision rule by Ma et al.	✓			✓	✓	✓
Halpern's rule	✓			✓		
Smets' rule of source-conditioning	✓	✓				✓
Smets' rule of data-conditioning	✓	✓		✓		✓
Our rule of inner revision	✓	✓	✓		✓	✓
Our rule of outer revision	✓	✓	✓	✓	✓	✓

Acknowledgements

The first author is partly supported by Key project for basic research from the Ministry of Science and Technology of China (Grant Number: 2012CB316205) and the third author is partially funded by the National Natural Science Foundation of China under Grant No. 61170012. We would like to thank Professor Jianbing Ma for many constructive discussions at different stages of this research.

References

- [BENFERHAT ET AL., 2011] BENFERHAT, S., TABIA, K., AND SEDKI, K. (2011). JEFFREY'S RULE OF CONDITIONING IN A POSSIBILISTIC FRAMEWORK - AN ANALYSIS OF THE EXISTENCE AND UNIQUENESS OF THE SOLUTION. *Ann. Math. Artif. Intell.*, 61(3):185–202.

- [CHAN AND DARWICHE, 2002] CHAN, H. AND DARWICHE, A. (2002). A DISTANCE MEASURE FOR BOUNDING PROBABILISTIC BELIEF CHANGE. IN DECHTER, R. AND SUTTON, R. S., EDITORS, *AAAI/IAAI*, PAGES 539–545. AAAI PRESS / THE MIT PRESS.
- [CHAN AND DARWICHE, 2003] CHAN, H. AND DARWICHE, A. (2003). ON THE REVISION OF PROBABILISTIC BELIEFS USING UNCERTAIN EVIDENCE. IN GOTTLÖB, G. AND WALSH, T., EDITORS, *IJCAI*, PAGES 99–105. MORGAN KAUFMANN.
- [DUBOIS AND PRADÉ, 1991] DUBOIS, D. AND PRADÉ, H. (1991). UPDATING WITH BELIEF FUNCTIONS, ORDINAL CONDITIONAL FUNCTIONS AND POSSIBILITY MEASURES. IN BONISSONE, P., M., H., L., K., AND J., L., EDITORS, *UAI*, PAGES 311–330. ELSEVIER.
- [DUBOIS AND PRADÉ, 1993] DUBOIS, D. AND PRADÉ, H. (1993). BELIEF REVISION AND UPDATES IN NUMERICAL FORMALISMS: AN OVERVIEW, WITH NEW RESULTS FOR THE POSSIBILISTIC FRAMEWORK. IN BAJCSY, R., EDITOR, *IJCAI*, PAGES 620–625. MORGAN KAUFMANN.
- [HALPERN, 2005] HALPERN, J. (2005). *Reasoning about Uncertainty*. MIT PRESS.
- [ICHIHASHI AND TANAKA, 1989] ICHIHASHI, H. AND TANAKA, H. (1989). JEFFREY-LIKE RULES OF CONDITIONING FOR THE DEMPSTER-SHAFER THEORY OF EVIDENCE. *Int. J. Approx. Reasoning*, 3(2):143–156.
- [JEFFREY, 1983] JEFFREY, R. (1983). *The Logic of Decision*. UNIVERSITY OF CHICAGO PRESS, SECOND EDITION.
- [JOUSSELME AND MAUPIN, 2012] JOUSSELME, A. AND MAUPIN, P. (2012). DISTANCES IN EVIDENCE THEORY: COMPREHENSIVE SURVEY AND GENERALIZATIONS. *Int. J. Approx. Reasoning*, 53(2):118–145.
- [MA ET AL., 2010] MA, J., LIU, W., DUBOIS, D., AND PRADÉ, H. (2010). REVISION RULES IN THE THEORY OF EVIDENCE. IN *ICTAI (1)*, PAGES 295–302. IEEE COMPUTER SOCIETY.
- [MA ET AL., 2011] MA, J., LIU, W., DUBOIS, D., AND PRADÉ, H. (2011). BRIDGING JEFFREY’S RULE, AGM REVISION AND DEMPSTER CONDITIONING IN THE THEORY OF EVIDENCE. *International J. on AI Tools*, 20 (4):691–720.
- [PEARL, 1988] PEARL, J. (1988). *Probabilistic Reasoning in Intelligent Systems - Networks of Plausible Inference*. MORGAN KAUFMANN SERIES IN REPRESENTATION AND REASONING. MORGAN KAUFMANN.
- [SHAFFER, 1976] SHAFFER, G. (1976). *A Mathematical Theory of Evidence*. PRINCETON UNIVERSITY PRESS, PRINCETON, N.J.
- [SHAFFER, 1981] SHAFFER, G. (1981). JEFFREY’S RULE OF CONDITIONING. *Philosophy of Sciences*, 48:337–362.
- [SMETS, 1993A] SMETS, P. (1993A). JEFFREY’S RULE OF CONDITIONING GENERALIZED TO BELIEF FUNCTIONS. IN HECKERMAN, D. AND MAMDANI, E. H., EDITORS, *UAI*, PAGES 500–505. MORGAN KAUFMANN.
- [SMETS, 1993B] SMETS, P. (1993B). QUANTIFYING BELIEFS BY BELIEF FUNCTIONS: AN AXIOMATIC JUSTIFICATION. IN BAJCSY, R., EDITOR, *IJCAI*, PAGES 598–605. MORGAN KAUFMANN.
- [WAGNER, 1992] WAGNER, C. G. (1992). GENERALIZING JEFFREY CONDITIONALIZATION. IN DUBOIS, D. AND WELLMAN, M. P., EDITORS, *UAI*, PAGES 331–335. MORGAN KAUFMANN.

AUAI Press
P.O. Box 866
Corvallis, Oregon 97339
USA